PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-283211
(43)Date of publication of application : 23.10.1998

(51)Int.Cl. G06F 9/46

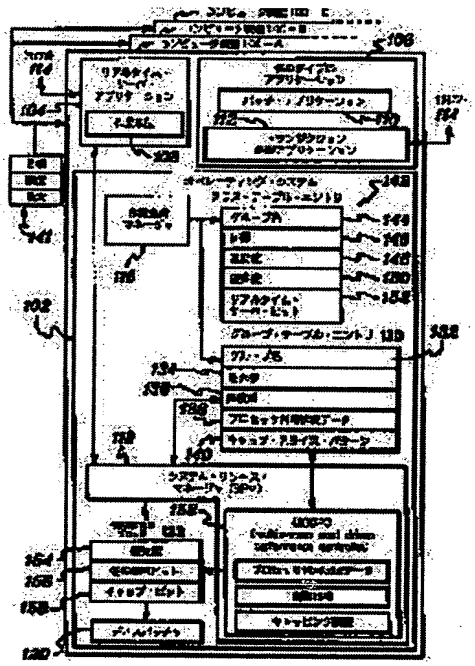
(21)Application number : 10-053575
(22)Date of filing : 05.03.1998

(71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
(72)Inventor : CATHARIN KRUGER AYRERT
PETER BURGERSEN WAKOM

(30)Priority
Priority number : 97 827529 Priority date : 28.03.1997 Priority country : US

(54) PROCESSOR RESOURCE MANAGEMENT METHOD FOR MULTI-SYSTEM ENVIRONMENT

(57)Abstract:
PROBLEM TO BE SOLVED: To simultaneously execute the application and work of another type on the same system by allocating the certain amount of processor resources to the real time application of a multi-system environment.
SOLUTION: A real time server application 104 distributes real time data and the application 106 of the other type is provided with a batch application 110 and a transaction processing application 112. A work load manager 116 manages the work loads of a computer system 100-A. For instance, the real time applications A and B are allocated to a resource group 1, a limit value is defined as 500, a maximum value is defined as 550, the real time application C is allocated to the resource group 2 and the limit value of 200 and the maximum value of 210 are taken. Similarly, non real time applications D and E are allocated to the resource group 3, the limit value of 2000 is defined and the maximum value is not used.



LEGAL STATUS

[Date of request for examination] 25.03.1999
[Date of sending the examiner's decision of rejection] 16.04.2002
[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]
[Date of final disposal for application]

(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

特開平10－283211

(43)公開日　平成10年(1998)10月23日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/46	３６０	Ｇ０６Ｆ　9/46	３６０Ｃ

審査請求　未請求　請求項の数22　ＯＬ　（全 28 頁）

(21)出願番号　特願平10－53575

(22)出願日　平成10年(1998)３月５日

(31)優先権主張番号　０８／８２７５２９

(32)優先日　1997年３月28日

(33)優先権主張国　米国（ＵＳ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

(72)発明者　キャサリン・クルーガー・アイラート
アメリカ合衆国12590、ニューヨーク州ワッピンガーズ・フォールズ、シェアウッド・ハイツ・ドライブ　34

(74)代理人　弁理士　坂口　博　（外１名）

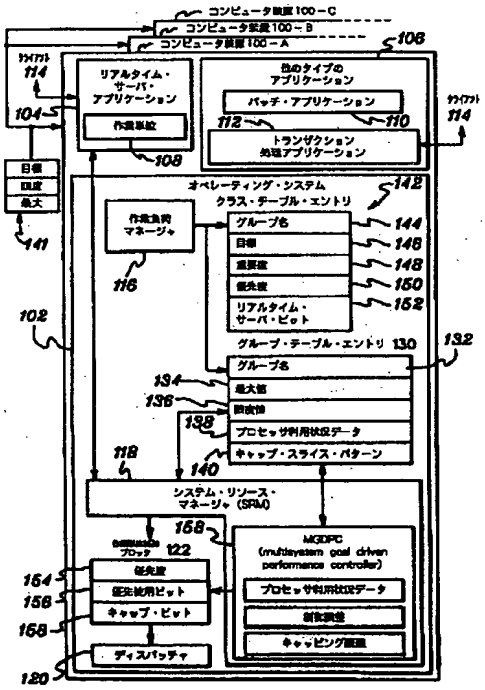
最終頁に続く

(54)【発明の名称】　マルチシステム環境のプロセッサ・リソース管理方法

(57)【要約】

【課題】　リアルタイム・データ・ストリームを割込みなくスムーズに配信する一方、システムで同時に実行される他の作業に所望の量のプロセッサ・リソースを提供する方法を提供する。

【解決手段】　リアルタイム・アプリケーションを実行するため、マルチシステム環境内のシステムが選択される。選択プロセスでは、リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するある量のプロセッサ・リソースが、選択されたシステムの他の作業に与える影響が最小のシステムが探される。リアルタイム・アプリケーションには、リアルタイム・アプリケーションのグループに選択された限度を超えない、ある量のプロセッサ・リソースが割当てられる。選択された量のプロセッサ・リソースは、マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できる。



1

【特許請求の範囲】

【請求項1】複数の装置で構成されたマルチシステム環境のプロセッサ・リソースを管理する方法であって、

少なくともリアルタイム・アプリケーションを含むリアルタイム・アプリケーションのグループに選択された限度を超えないある量のプロセッサ・リソースを前記マルチシステム環境のリアルタイム・アプリケーションに割当て、選択された量の前記プロセッサ・リソースは、前記マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できるステップと、

前記複数の装置のうち、選択された装置で前記リアルタイム・アプリケーションを処理するステップと、

を含む、方法。

【請求項2】前記装置を選択するステップを含み、前記選択ステップは、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するある量の前記プロセッサ・リソースが、選択された装置の他の作業に与える影響は最小である装置を複数の装置のうち少なくとも２つの装置から選択するステップを含む、請求項1記載の方法。

【請求項3】前記割当てステップは、

少なくとも２つの装置から、リアルタイム・データ・レートを配信するための、所与の時間内での、最大サービス・コストを求めるステップと、

前記リアルタイム・データ・ストリームを配信するサービス・レートを、前記最大サービス・コストを使用して計算するステップと、

前記サービス・レートが前記限度内にあるかどうか確認するステップであって、前記選択ステップは、前記サービス・レートが前記限度内にあるときに実行される、前記確認するステップと、

を含む、請求項2記載の方法。

【請求項4】前記確認ステップは、

少なくとも２つの装置の高サービス・コストを確認するステップと、

前記最大サービス・コストを取得するため、前記少なくとも２つの高サービス・コストを比較するステップと、

を含む、請求項3記載の方法。

【請求項5】少なくとも選択されていない任意の装置に、選択された装置について通知するステップを含む、請求項2記載の方法。

【請求項6】前記通知ステップは、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するために計算されたサービス・レートの指示を、前記少なくとも選択されていない任意の装置に送るステップを含む、請求項5記載の方法。

【請求項7】前記処理ステップは、リアルタイム・アプリケーションの前記グループに選択された前記プロセッサ・リソースの最大量を、前記リアルタイム・アプリケーションが超えるのを防ぐステップを含む、請求項1記載の方法。

2

【請求項8】前記防止ステップは、前記リアルタイム・アプリケーションの作業単位が前記最大量を超えずにはディスパッチ不能であることを示すステップを含む、請求項7記載の方法。

【請求項9】前記リアルタイム・アプリケーションの処理が求められていることを、前記複数の装置のいくつかに示すステップを含む、請求項1記載の方法。

【請求項10】前記指示ステップは、

前記リアルタイム・アプリケーションに選択されたディスパッチ優先度を前記いくつかの装置内に設定するステップと、

前記ディスパッチ優先度は調整されないことを、前記いくつかの装置内に指定するステップと、

を含む、請求項9記載の方法。

【請求項11】前記複数の装置が、前記限度を利用できるようにするステップを含む、請求項1記載の方法。

【請求項12】複数の装置で構成されたマルチシステム環境のプロセッサ・リソースを管理するために、コンピュータ読取り可能プログラム・コード手段が具体化されたコンピュータ使用可能媒体を含む製品であって、前記コンピュータ読取り可能プログラム・コード手段は、

少なくともリアルタイム・アプリケーションを含むリアルタイム・アプリケーションのグループに選択された限度を超えない、ある量のプロセッサ・リソースを前記マルチシステム環境のリアルタイム・アプリケーションに割当てることを、コンピュータによって可能にし、選択された量の前記プロセッサ・リソースは前記マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できる、コンピュータ読取り可能プログラム・コード手段と、

前記複数の装置のうち選択された装置での前記リアルタイム・アプリケーションの処理をコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、

を含む、製品。

【請求項13】前記装置の選択をコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含み、コンピュータによって選択を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するある量の前記プロセッサ・リソースが、選択された装置の他の作業に与える影響は最小である装置を、複数の装置のうち少なくとも２つの装置から選択することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項12記載の製品。

【請求項14】コンピュータによって割当てを可能にする、前記コンピュータ読取り可能プログラム・コード手

3

段は、
リアルタイム・データ・レートを配信するため、所与の時間内の最大サービス・コストを少なくとも２つの装置から求めることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
前記リアルタイム・データ・ストリームを配信するサービス・レートを、前記最大サービス・コストを使用して計算することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
前記サービス・レートが前記限度内にあるかどうか確認することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段であって、前記選択ステップは前記サービス・レートが前記限度内にあるとき実行される、前記コンピュータ読取り可能プログラム・コード手段と、
を含む、
請求項１３記載の製品。
【請求項１５】コンピュータによって確認を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、
少なくとも２つの装置の高サービス・コストを求めることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
前記最大サービス・コストを取得するため、前記少なくとも２つの高サービス・コストを比較することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
を含む、請求項１４記載の製品。
【請求項１６】少なくとも選択されていない任意の装置に、選択された装置について通知することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１３記載の製品。
【請求項１７】コンピュータによって通知を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するために計算されたサービス・レートの指示を、少なくとも選択されていない任意の装置に送ることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１６記載の製品。
【請求項１８】コンピュータによって処理を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、リアルタイム・アプリケーションの前記グループに選択された前記プロセッサ・リソースの最大量を前記リアルタイム・アプリケーションが超えるのを防ぐことをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１２記載の製品。
【請求項１９】コンピュータによって防止を可能にする、前記コンピュータ読取り可能プログラム・コード手

4

段は、前記リアルタイム・アプリケーションの作業単位が前記最大量を超えずにはディスパッチ不能であることを指示することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１８記載の製品。
【請求項２０】前記リアルタイム・アプリケーションの処理が求められることを前記複数の装置のうちいくつかの装置に示すことをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１２記載の製品。
【請求項２１】コンピュータによって指示を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、
前記リアルタイム・アプリケーションに選択されたディスパッチ優先度を前記いくつかの装置内に設定することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
前記ディスパッチ優先度は調整されないことを前記いくつかの装置内に指定することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、
を含む、請求項２０記載の製品。
【請求項２２】前記複数の装置が前記限度を利用できるようにすることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、請求項１２記載の製品。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、一般的には、マルチシステム環境のプロセッサ・リソース管理に関し、特にマルチシステム環境から、割込みのないスムーズなリアルタイム・データ・ストリームを提示するのに適したプロセッサ・リソースを提供でき、同じシステム上の他のタイプのアプリケーションや作業を同時に実行できるシステムを選択することに関する。
【０００２】
【従来の技術】コンピュータ環境では、質のよいリアルタイム・データ・ストリームを配信することは大きな課題である。リアルタイム・データ・ストリームには、例えば、ビデオ、オーディオ、及びマルチメディア等のデータ・ストリームが含まれる。現在、リアルタイム・データ・ストリームを配信するためのアプローチは、構成管理と高ディスパッチ優先度の２つがある。それぞれについて説明する。
【０００３】構成管理アプローチでは、リアルタイム・データ・ストリームの配信にサーバが専用される。特にサーバは、リアルタイム・データ・ストリームを所与のレートで配信するために、充分な容量を持たせて構成される。サーバはリアルタイム・データ・ストリームの配信専用なので、リアルタイム・ストリームはシステム・

5

リソースを独占し、他のタイプの作業については、残るとしてもごくわずかなリソースしか残らない。

【０００４】高ディスパッチ優先度アプローチの場合、リアルタイム・アプリケーションはディスパッチ優先度がかなり高い状態で実行される。この手法は専用システムで実行する必要はないが、この場合でも、リアルタイム・アプリケーションをかなり高い優先度で実行することによって、実行に充分なリソースをシステムの他の作業が受け取ることに対する制御あるいは保証はない。

【０００５】

【発明が解決しようとする課題】従って、リアルタイム・データ・ストリームを割込みなくスムーズに配信する方、システムで同時に実行される他の作業に所望の量のプロセッサ・リソースを提供する方法が求められる。またリアルタイム・データ・ストリームに割当てられるリソースの量に限度を設け、これにより、選択されたリソース量を非リアルタイム作業に利用できる管理方法も求められる。さらに、リアルタイム・アプリケーション、及びシステムの他の非リアルタイム作業の必要性を最適な形で満足するシステムを、複数のシステムの中から選択できるメカニズムが求められる。

【０００６】

【課題を解決するための手段】従来技術の欠点は、複数のシステムで構成されるマルチシステム環境のプロセッサ・リソースを管理する方法を通して克服され、他の利点も与えられる。方法は、例えば、ある量のプロセッサ・リソースをマルチシステム環境のリアルタイム・アプリケーションに割当てるステップを含む。この量は、１つ以上のリアルタイム・アプリケーションのグループのために選択された限度を超えることはない。また、選択された量のプロセッサ・リソースは、マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できる。また方法は、複数のシステムの中から選択されたシステムでのリアルタイム・アプリケーションの処理を含む。

【０００７】例えば、この方法はさらに、システムを選択するステップを含む。選択ステップは、リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するためのある量のプロセッサ・リソースが、選択されたシステムの他の作業に与える影響は最小のシステムを複数のシステムのうち少なくとも２つのシステムから選択するステップを含む。

【０００８】他の例では、割当てステップはさらに、少なくとも２つのシステムから、リアルタイム・データレートを配信するための、所与の時間内で最大のサービス・コストを求めるステップ、リアルタイム・データ・ストリームを配信するサービス・レートを、最大サービス・コストを使用して計算するステップ、及びサービス・レートが限度内にあるかどうか確認するステップを含む。サービス・レートが限度内にあれば選択操作が実行

6

される。

【０００９】他の例の場合、この方法は、少なくとも選択されていない任意のシステムに、選択されたシステムについて通知するステップを含む。

【００１０】本発明の他の側面では製品(article of manufacture) が提供される。製品には、コンピュータ読取り可能プログラム・コード手段を有するコンピュータ使用可能媒体が含まれる。プログラム・コード手段は、複数のシステムを含むマルチシステム環境のプロセッサ・リソースを管理するために実現される。１つの例では、製品のコンピュータ読取り可能プログラム・コード手段に、マルチシステム環境のリアルタイム・アプリケーションにある量のプロセッサ・リソースをコンピュータが割当てるようにする、コンピュータ読取り可能プログラム・コード手段が含まれる。この量は、少なくともリアルタイム・アプリケーションを含む、１つ以上のリアルタイム・アプリケーションのグループのために選択された限度を超えることはない。選択された量のプロセッサ・リソースは、マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できる。製品はさらに、複数のシステムのうち選択されたシステムでリアルタイム・アプリケーションをコンピュータが処理するようにする、コンピュータ読取り可能プログラム・コード手段を含む。

【００１１】本発明の管理機能の利点は、リアルタイム・アプリケーションを実行しながら、リアルタイム・アプリケーションがシステム全体の制限を超過させるのを防ぐ、マルチシステム環境のシステムを選択できることである。選択プロセスは、システムの重要度が最小の作業の置き換えを考慮している。一部の作業は置き換えることができるが、本発明では、リアルタイム・アプリケーションにより利用されるある量のプロセッサ・リソースを引き続き選択でき、選択された量のリソースを非リアルタイム・アプリケーションのために予約できる。本発明に従って、リアルタイム・アプリケーションは割込みなくスムーズに実行でき、システムの他のタイプの作業も処理できる。

【００１２】

【発明の実施の形態】本発明に従い、リアルタイム・アプリケーション、及びバッチ処理、トランザクション処理等、他のタイプのアプリケーションも、汎用マルチシステム環境(つまり非専用コンピュータ装置) で同時に実行できる。特にリアルタイム・アプリケーションには、リアルタイム・アプリケーションがシステムの制限を超過させないように、一定量のプロセッサ・リソースが与えられる。他のタイプのアプリケーション(つまり非リアルタイム・アプリケーション) にも、ある量のプロセッサ・リソースが処理のため与えられる。非リアルタイム・アプリケーションは、ここで用いているとおり、例えば、バッチ・アプリケーション、トランザクシ

7

ョン処理アプリケーション、非リアルタイムにシステムで実行される他の作業等、リアルタイムに実行されない任意のアプリケーションを含む。

【００１３】本発明に従って、リアルタイム・アプリケーションを実行するためにマルチシステム環境の1つのシステムが選択される選択プロセスが提供される。選択プロセスは、マルチシステム環境内の重要度が最小の作業の置き換えを考慮する。

【００１４】本発明の機能を取り入れて使用するマルチシステム環境の1つの例を図1に示し、ここで説明する。

【００１５】ある実施例でマルチシステム環境は、複数のコンピュータ装置(コンピュータ装置１００－Ａ、１００－Ｂ、１００－Ｃ等)を含む。システムは、例えば、相互接続され、互いに連係し、独立したコンピュータ装置である。当業者には明らかなように、このような相互接続され連係するコンピュータ装置は、本発明の主旨から逸脱することなく任意個数を使用できる。

【００１６】このようなコンピュータ装置の1つ(コンピュータ装置１００－Ａ等)についてここで詳しく説明する。ただしマルチシステム環境のどのコンピュータ装置でも、ここで説明しているコンポーネントを追加できることは理解されよう。

【００１７】1つの例では、コンピュータ装置１００－Ａに、例えば、オペレーティング・システム１０２、少なくとも１つのリアルタイム・サーバ・アプリケーション１０４、及び他のタイプのアプリケーション１０６が含まれる。リアルタイム・サーバ・アプリケーション１０４はリアルタイム・データを配信し、作業単位１０８を含む。作業単位は、例えば、コンピュータ装置の目的である有益な作業を行うアプリケーション・プログラムである。他のタイプのアプリケーション１０６は、例えば、バッチ・アプリケーション１１０とトランザクション処理アプリケーション１１２を含む。これも作業単位を含む。リアルタイム・サーバ・アプリケーションとトランザクション処理アプリケーションは、例えば、コンピュータ装置１００に接続されたワークステーション側のサービング・クライアント１１４である。

【００１８】オペレーティング・システム１０２は、1つの例として、International Business Machines CorporationのＭＶＳ(多重仮想記憶)オペレーティング・システムである。ただしこれは1つの例にすぎない。他のオペレーティング・システムも本発明の主旨または範囲から逸脱することなく使用できる。ある実施例では、オペレーティング・システム１０２は作業負荷マネージャ１１６、システム・リソース・マネージャ１１８、ディスパッチャ１２０、及び作業単位制御ブロック１２２を含む。以下、それぞれについて詳しく説明する。

【００１９】作業負荷マネージャ１１６はコンピュータ装置１００－Ａの作業負荷を管理する。作業負荷マネージャのタスクの1つは、本発明の管理機能によって用いられるある値を入力としてコンピュータ装置に読込むことを含む。例えば、作業負荷マネージャはプロセッサ消費限度値とプロセッサ消費最大値を読込む。これらの値は、リアルタイム・アプリケーション(つまりリアルタイム・データを配信するアプリケーション)及び非リアルタイム・アプリケーションに割当てられるリソースの量を確認するために用いられる。１つの例として、これらの値はシステム管理者によって設定される。

【００２０】プロセッサ消費限度値はそれぞれ、例えばリアルタイム・アプリケーションに割当てることのできるプロセッサ・リソースの量を指定する。この値は、システムの他のタイプのアプリケーションや作業が同時に実行できるように選択される。後述するように、異なるリアルタイム・アプリケーションを異なる限度値に関連付ける、または同じ限度値にすることができる。同様に、プロセッサ消費限度値は非リアルタイム・アプリケーションに関連付けることができる。

【００２１】プロセッサ消費最大値はそれぞれ、例えばリアルタイム・アプリケーション、または非リアルタイム・アプリケーションによって使用できるプロセッサ・リソースの量に対するキャップを指定する。ある実施例で、これらの値は限度値よりも大きい。これは、処理の間に、データ・ストリームの配信に一時的なサージがあり得、これにより割当て済みリソースの量が不足するとされるからである。従って、最大値はそれぞれ、処理側アプリケーションが、プロセッサ消費限度値により指定されたものより少し多くプロセッサを消費する必要のある場合を考慮してバッファを与える。アプリケーションによって消費されるリソースの量はこのキャップを超えない。これはコンピュータ装置１００－Ａに追加される作業に、選択された量のプロセッサ・リソースが与えられることを保証する。後述するように、異なるアプリケーションを異なる最大値に関連付けることができ、または同じ最大値にすることができる。

【００２２】ある実施例で、プロセッサ消費限度値と最大値はプロセッサ・サービス単位として表される。ただしこれは1つの例にすぎない。当業者には明らかなように、他の測定単位も、本発明の主旨または範囲から逸脱することなく選択できる。

【００２３】作業負荷マネージャは、限度値と最大値の読込みに続いて、各プロセッサの消費限度値と消費最大値のペアを取り、アプリケーションの個々の作業単位が割当てられるリソース・グループを設定する。これにより、異なる限度値と最大値を異なるアプリケーションに指定できる。例えば、リアルタイム・アプリケーションＡ、Ｂをリソース・グループ1に割当て、限度値を５００、最大値を５５０とすることができる。また、リアルタイム・アプリケーションＣがリソース・グループ2に割当てられ、限度値２００、最大値２１０が取られる。

9

同様に、非リアルタイム・アプリケーションD、Eがリソース・グループ3に割当てられ、限度値2５００とされ、最大値は使用されない。これらの例で、リアルタイムと非リアルタイムのアプリケーションは別々のグループである。これは1つの例にすぎない。他の実施例で、グループにタイプの異なるアプリケーションを追加することもできる。限度値と最大値はグループに対して選択される。従ってグループは全体としてこれらの値を超えることはない。

【００２４】各リソース・グループは、リソース・グループ・テーブル・エントリ１３０によってオペレーティング・システムに割当てられたメモリで表される。ある実施例で、リソース・グループ・テーブル・エントリ１３０は次を含む。

（ａ）リソース・グループ名１３２。作業負荷マネージャへの入力値であり、このリソース・グループ・エントリに対応したリソース・グループの名前を示す。

（ｂ）プロセッサ消費最大値１３４。これは先に述べたように、このリソース・グループのアプリケーションによって消費されることが許可されたプロセッサ・リソースの量に対するキャップを指定する入力値である。

（ｃ）プロセッサ消費限度値１３６。これは先に述べたように、このグループのアプリケーションに割当てることのできるプロセッサ・リソースの量を指定する入力値である。

（ｄ）プロセッサ利用状況データ１３８。これは、後述するさまざまな測定データ及び計算データを含む。

（ｅ）キャップ・スライス・パターン１４０。これは、詳細は後述するが、リソース・グループの作業単位にキャップを付けるときの特定タイム・スライスを示す計算値である。

【００２５】前記の他に、作業負荷マネージャ１１６は、相対的重要度を含めて、コンピュータ装置１００－Ａの性能目標を入力として読込み、個々の作業単位（例えば、リソース・グループの作業単位）が割当てられるユーザ性能目標クラスを設定する。性能目標は、リアルタイム及び非リアルタイムの両方のアプリケーションに関連付けられる。目標クラスがリソース・グループに関連付けられる可能性は五分五分である。リソース・グループの一部ではない作業単位も目標クラスの一部になり得る。また、クラスは2つ以上をリソース・グループに関連付けることができる。

【００２６】本発明に従って、プロセッサ消費限度値、プロセッサ消費最大値、及び目標がマルチシステム環境全体の値及び目標になる。つまりマルチシステム環境の、本発明の管理機能を担うどのコンピュータ装置によっても、同じ値と目標が用いられる。値と目標は、例えば、マルチシステム環境のコンピュータ装置に接続された外部記憶装置１４１（ＤＡＳＤ等）に格納されるので、情報が必要なシステムは情報をすぐに取得できる。

10

【００２７】性能目標とユーザ性能目標のクラスについては、J. D. Amanらによる１９９５年１２月５日付米国特許番号第５４７３７７３号、"Apparatus and Method forManaging A Data Processing System Workload According to Two or More Distinct Processing Goal Types"、及びC. K. Eilertらによる１９９５年２月３日付米国特許出願番号第０８／３８３０４２号、"Multi-System Resource Capping"を参照されたい。

【００２８】ユーザ性能目標クラスはそれぞれ、クラス・テーブル・エントリ１４２によってオペレーティング・システムに割当てられたメモリで表される。クラス・テーブル・エントリ１４２は、例えば次を含む。

（ａ）リソース・グループ名１４４。これはリソース・グループがあると仮定して、ユーザ性能目標クラスが属するリソース・グループを指定する入力値である。リソース・グループがなければ、このフィールドはブランクのままである。

（ｂ）ユーザ性能目標１４６。これはクラスの特定の性能目標を示す入力値である。性能目標の例として所望の応答時間がある。

（ｃ）ユーザ性能目標の相対重要度１４８。これもコンピュータ装置に対するこの目標の重要度を指定する入力値である。

（ｄ）ディスパッチ優先度１５０。これはコントローラ１５８によってセットされる。このクラスの作業単位が（全体として）ディスパッチされる順序を、システムの他のクラスに対して示す。

（ｅ）リアルタイム・サーバ・ビット１５２。これはこのクラスの作業単位がリアルタイム・サービング・アプリケーションの一部かどうかを示す。このビットは、後述するように本発明に従ってセットされる。

【００２９】オペレーティング・システム１０２は、作業負荷マネージャの他にシステム・リソース・マネージャ１１８を含む。システム・リソース・マネージャはオペレーティング・システムのコンポーネントであり、コンピュータ装置のリソース（ＣＰＵ利用率等）を管理する。ある実施例でシステム・リソース・マネージャ１１８はMGDPC（multisystem goal driven performance controller）１５８を含む。MGDPCは、後述するように、オペレーティング・システムに適応性と自己調整性を持たせるよう、性能問題の増分検出と補正を行うため、フィードバック・ループを与える。

【００３０】MGDPCの実施例は、J. D. Amanらによる１９９５年１２月５日付米国特許番号第５４７３７７３号、"Apparatus and Method for Managing A Data Processing System Workload According To Two or More Distinct Processing Goal Types"、C. K. Eilertらによる１９９５年２月３日付米国特許出願番号第０８／３８３０４２号、"Multi-System Resource Capping"、及びC. K. EilertとP. Yocomによる１９９５年２月３日付

米国特許出願番号第０８／３８３１６８号、"Apparatus and Method for Managing A Distributed Data Processing System Workload According To A Plurality of Distinct Processing Goal Types"を参照されたい。

【００３１】コントローラ１５８の、特に本発明に関わる他の詳細について、ここで詳しく説明する。コントローラ１５８は、本発明に従って、本発明のプロセッサ利用状況データの処理、ユーザ性能目標の達成を管理するための性能インデックスの計算とシステム・リソース管理の調整、及びリアルタイム・アプリケーションがシステムの制限を超過させないためのプロセッサ・リソース消費量のキャッピングの調整を担当する。以下、それぞれについて説明する。

【００３２】オペレーティング・システム１０２は、さらにディスパッチャ１２０を含む。ディスパッチャ１２０は、コンピュータ装置１００－Aによって次に実行される作業単位を選択するために用いられる。各作業単位は作業単位制御ブロック（WUCB）１２２の1つによって表される。作業単位制御ブロックはそれぞれオペレーティング・システム１０２に割当てられたメモリに格納され、例えば、次の3つのディスパッチ制御フィールドを含む。

（ａ）ディスパッチ優先度フィールド１５４。これはディスパッチャがディスパッチ作業に取り掛かる順序をディスパッチャ１２０に示す。ディスパッチ優先順位の1つの例を図2に示す。図2に示すように、ある実施例で最大のディスパッチ優先度はオペレーティング・システム、ディスパッチャ、システム・リソース・マネージャ、及びMGDPCに予約され、次に最大の優先度は、リアルタイム・アプリケーションの作業単位及びシステム・ファンクションのために使われる。トランザクション及びバッチ処理のアプリケーションの作業単位には複数の低い優先度が予約される。これら複数の低優先度は、先に述べた特許及び特許出願に詳しく述べられているとおり、MGDPCの操作によって割当てられる。

（ｂ）優先使用ビット・フィールド１５６。これは関連付けられた作業単位を、レディ状態になってすぐにディスパッチ対象として考慮するかどうかをディスパッチャに示す。

（ｃ）キャップ・ビット・フィールド１５８。これは関連付けられた作業単位をディスパッチできるかどうかをディスパッチャに示す。このビットは、ここで詳しく述べているように、作業単位の特定のグループのために許可されたプロセッサ実行時間の量を制御するために用いられる。

【００３３】本発明に従って、選択された量のプロセッサ・リソース（ＣＰＵ時間等）をリアルタイム・アプリケーションに、また選択された量をシステムの他のタイプの作業に提供するため、コンピュータ装置のさまざまなコンポーネントが用いられる。本発明に従って、リアルタイム・アプリケーションの処理に関連付けられるロジックの実施例について、図3を参照して詳しく説明する。

【００３４】図3を参照する。最初、実行したいリアルタイム・アプリケーションが、アプリケーションを実行する可能性のあるそれぞれのシステム上のシステム・リソース・マネージャ１１８に対して自身を識別する（ステップ３０２）。（これらのシステムはそれぞれ、リアルタイム・アプリケーションのコピーを含んでいる。）これによりリソース・マネージャは、リアルタイム・アプリケーションが割込みなくスムーズに処理を行う一方、システムの他の作業の制限を超過させないため充分なリソースを持つのに必要な調整を行うことができる。システム・リソース・マネージャに対してアプリケーションを識別する手順の1つを、図4を参照して詳しく説明する。図4に示したファンクションは、1つの例ではシステム・リソース・マネージャ１１８によって実行される。

【００３５】図4を参照する。１つの例として、最初にクラス・テーブル・エントリ１４２のディスパッチ優先度１５０と、識別側リアルタイム・アプリケーションに関連付けられた各作業単位制御ブロック１２２の優先度１５４が"次に最大の優先度"に設定される。これはリアルタイム・アプリケーションのために指定される（ステップ４０２）。

【００３６】その後、リアルタイム・アプリケーションのための個々の作業単位制御ブロック１２２の各優先使用ビット１５６がセットされ、ディスパッチについて関連付けられた作業単位を、利用できるようになればすぐに考慮すべきことがディスパッチャに示される（ステップ４０４）。

【００３７】また、クラス・テーブル・エントリ１４２のリアルタイム・サーバ・ビット１５２がセットされ、クラスの作業単位がリアルタイム・サーバであり、ディスパッチ優先度が調整されないことがMGDPCコントローラ１５８に示される（ステップ４０６）。

【００３８】ディスパッチ優先度、全優先使用ビット、及びリアルタイム・サーバ・ビットのセット後、識別ファンクションは完了する。従って、図3に戻るが、リアルタイム・アプリケーションの次のタスクは、リソースをアプリケーションに割当てることである（ステップ３０４）。

【００３９】割当て方法の実施例を図5乃至図9を参照して詳しく説明する。図6乃至図9のロジックはさまざまなプロセッサ利用状況データを使用する。プロセッサ利用状況データについては図5を参照して詳しく説明する。データの説明の後、割当てロジックの実施例について説明する。

【００４０】図5を参照する。本発明のプロセッサ利用状況データ１３８の1つの例は次を含む。

13

（ａ）グループ・サービス・レート５０２。これはグループの作業単位によって消費された現在のサービス（ＣＰＵ消費等）を示す。このレートは普通はサービス単位／秒で表されるが、これは必要ではない。

（ｂ）前の割当て済みリアルタイム・レート５０４。これは性能コントローラが起動されたとき（例えば１０秒毎）ＭＧＤＰＣによって埋められる。性能コントローラは現在の割当て済みリアルタイム・レート（後述）を前の割当て済みリアルタイム・レートにコピーする。

（ｃ）現在の割当て済みリアルタイム・レート５０６。これはグループのリアルタイム・データ・ストリームを配信するために現在割当てられている容量単位の量（ＫＢ／秒等）を表す。ディスクから通信ネットワークへデータのバイトを送るプロセッサ・コストは、容量単位としてＫＢ／秒を使用する定数に充分近く、ＫＢ／秒はリアルタイム・サービング・アプリケーションには有意義な単位である。リアルタイム・サービング・アプリケーションは容量をＫＢ／秒で要求する。現在割当てられているリアルタイム・レートは、次に述べる割当てファンクションと割当て解除ファンクションによって埋められる。

（ｄ）前の一時停止リアルタイム・レート５０８。これはコントローラ１５８が起動したとき（例えば１０秒毎）ＭＧＤＰＣコントローラ１５８によって埋められる。コントローラ１５８は現在の一時停止リアルタイム・レート（後述）を前の一時停止リアルタイム・レートのフィールドにコピーする。

（ｅ）現在の一時停止リアルタイム・レート５１０。これはグループの一時停止した任意のリアルタイム・データ・ストリームのデータ・レートの和を含む（後述）。

（ｆ）割当て済みリアルタイム・レート・ウィンドウ５１２。これは、例えば１８のウィンドウ枠を含むデータのウィンドウである。特に割当て済みリアルタイム・レートは、移動するタイム・ウィンドウで維持される。移動するタイム・ウィンドウは、例えば最近の３分として定義される。１８０秒のウィンドウは、それぞれが１０秒である１８のインターバル（１８のウィンドウ枠という）に分けられ、１８の要素配列として実現される。割当て済みリアルタイム・レート・ウィンドウ枠は、コントローラ１５８が１０秒毎に起動されたとき、前の１０秒間隔の間にリアルタイム・サービング・アクティビティがあった場合はコントローラ１５８によって埋められる。割当て済みリアルタイム・ウィンドウ枠は、前の割当て済みリアルタイム・レートと現在の割当て済みリアルタイム・レートの平均を取ることによって計算される。

（ｇ）リアルタイム・サービス・レート・ウィンドウ５１４。もう１つのデータ・ウィンドウであり、例えば１８のウィンドウ枠を含む。リアルタイム・サービス・レートは、リアルタイム・データ・ストリームを配信するリアルタイム・アプリケーションの作業単位によって消費されたプロセッサ・サービスを表す。リアルタイム・サービス・レート・ウィンドウ枠は、コントローラ１５８が１０秒毎に起動されたとき、前の１０秒間隔の間にリアルタイム・サービング・アクティビティがあった場合に、コントローラ１５８によって埋められる。リアルタイム・サービス・レート・ウィンドウ枠の値はグループ・サービス・レートのフィールドからコピーされる。

（ｈ）リアルタイム一時停止レート・ウィンドウ５１６。これは第3のデータ・ウィンドウで、これも例えば１８のウィンドウ枠を含む。リアルタイム一時停止レートは、現在一時停止しているこのグループのリアルタイム・データ・ストリームを表すデータ・レートである。他の2つのデータのウィンドウと同様に、一時停止レート・ウィンドウ枠は、コントローラ１５８が１０秒毎に起動されるとき、前の１０秒間隔の間にリアルタイム・サービング・アクティビティがあった場合は、コントローラ１５８によって埋められる。一時停止リアルタイム・レート・ウィンドウ枠の値は、前の一時停止リアルタイム・レートと現在の一時停止リアルタイム・レートの平均を取ることによって計算される。

（ｉ）リアルタイム・ウィンドウ枠インデックス５１８。これはウィンドウ枠配列要素の現在のインデックスを表す。現在の１０秒間隔のデータは、現在のウィンドウ枠インデックスによってインデックスが付けられたウィンドウ枠配列要素に格納される。各ウィンドウ配列の１８の配列要素がローテーション方式で用いられる。１８個目の要素が用いられた後、次の時間間隔のデータが第1要素に格納され、その要素の古いデータと置き換えられる。リアルタイム・ウィンドウ枠インデックスは、前の１０秒間隔の間にリアルタイム・サービング・アクティビティがあった場合に、コントローラ１５８によって進められる。リアルタイム・サービング・アクティビティがないときはインデックスは進められない。これにより、再びリアルタイム・サービング・アクティビティが開始されたときに用いられるよう、ＫＢ／秒当たりの前のサービス・コストが保存される。

【００４１】先に述べたとおり、前記のプロセッサ利用状況データは、本発明の割当てファンクションの実行時に用いられる。一般的に、システム・リソース・マネージャの割当てファンクションは、リアルタイム・データ・ストリームを配信するためのプロセッサ容量を割当て、保証するために、リアルタイム・サーバによって起動される。割当てファンクションの入力値は、リアルタイム・ストリームのＫＢ／秒データ・レートである。割当てのとき、マルチシステム環境のシステムのＫＢ／秒当たり最大のプロセッサ・サービス・コストが確認される。次にこの値により、新しいストリームのサービス要件が確認される。その後、システム管理者によって指定されたグループ・サービス限度１３６の範囲内で、必要

なプロセッサ容量が利用できるかどうか確認される。割当てファンクションの実施例の詳細については、さらに図6を参照して説明する。

【００４２】図6を参照する。最初、マルチシステム・データが検索され、リアルタイム・データ・ストリームを配信するため、マルチシステム環境のシステムについてＫＢ当たり最大のサービス・レートが見つけられる（ステップ６００）。１つの例として、マルチシステム・データはシステム・データ・テーブルに格納され、システム・データ・テーブルはオペレーティング・システムに割当てられたメモリに置かれる。

【００４３】システム・データ・テーブルのエントリ７００の実施例を図7に示し、ここで詳しく説明する。１つの例として、マルチシステム環境の各システムについてシステム・データ・テーブルにエントリが存在し、各エントリは次の情報を含む。

（ａ）システム名７０２。これはこのエントリが属するシステムの名前を識別する。

（ｂ）ウィンドウの高サービス単位／ＫＢレート７０４。これはあるシステムの１８個の枠のウィンドウでのＫＢ／秒レート当たり最大のサービス単位である。この計算方法の1つの例については、ステップ１２００を参照して説明する（図１２）。

（ｃ）計算済みサービス７０６。これはコントローラ１５８の最後の定期的（１０秒等）起動の後、開始されたデータ・ストリームによって用いられる、計算されたサービスを表す。目的は、各システムで表示されると予測されるサービスが各システムで更新されて表示されるよう、割当て済みサービスを仮に記録しておくことである。これにより、１０秒間隔の間に限度を超えた割当てが防止され、コンピュータ装置の利用可能な容量について最新情報が与えられる。これを計算する方法の1つの例をここで説明する。

（ｄ）利用可能サービス配列７０８。これは重要度毎の利用可能サービス、及び未使用サービスを示す。１つの例では、次に示すように、各重要度（重要度0～6等）のエントリ、及び未使用サービスのエントリが配列に含まれる。

```
array element 1 service avail. at importance 0
array element 2 service avail. at importance 1
array element 3 service avail. at importance 2
array element 4 service avail. at importance 3
array element 5 service avail. at importance 4
array element 6 service avail. at importance 5
array element 7 service avail. at importance 6
array element 8 unused service
```

ここで述べている1つの例では、最大重要度は重要度0で、最小重要度は重要度7である。例えば、サービス・データは、次に詳しく示すように、６枠のウィンドウで計算され平均が取られる。

（ｅ）単一エンジンＣＰＵ速度７１０。これはエンジン当たりのＣＰＵ速度を毎秒サービス単位数で表す。

【００４４】図6を参照する。システム・データ・テーブルですべてのシステムについて最大レートが求められた後、グループ・サービス限度１３６内で所要プロセッサ容量（以下で計算）が利用できるかどうか確認される（ステップ６０２）。このチェックを行う実施例については、図8を参照して詳しく説明する。

【００４５】最初、新しいリアルタイム・データ・ストリームを配信するために必要なサービスが次のように計算される（ステップ８００）。

【数1】新しいストリームのＫＢ×ウィンドウのＫＢ当たりの最大サービス・レート

【００４６】ここで新しいストリームのＫＢは入力値、最大サービス・レートは、システム・データ・テーブルで確認されたレートである。

【００４７】必要なサービスの量（つまりサービス・レート）が計算された後、このサービス量がグループ・サービス限度１３６で利用できるかどうか確認される（照会８０２）。充分な容量が利用できない場合は、不成功を示す戻りコードがセットされ（ステップ８０４）、限度チェック・ファンクション及び割当てファンクションは完了する（ステップ８０６）。

【００４８】しかしリアルタイム・データ・ストリームを配信するのに充分な容量がある場合は、限度チェック・ファンクションは完了する（ステップ８０６）。

【００４９】図6に戻る。リアルタイム・データ・ストリームを配信するのに充分な容量がある場合、割当ての次のステップは、割当てられたリクエストをリアルタイム・サーバ・アプリケーションが実行するシステムを選択することである（ステップ６０４）。先に述べたように、割当てファンクションでは、限度内に充分な容量があるかどうかチェックされるほか、本発明に従って、新しいリクエストを実行すべきシステムが決定される。

【００５０】例えば、割当てるべき入力があり、これは新しいリクエストを開始できる可能性のあるシステムのリストを示す。例えば、要求される可能性のあるすべてのビデオを格納したシステムに、すべてのシステムが接続されているとは限らない。システムＡ、Ｂ、及びＣではビデオ（例えばディスク上）にアクセスできても、システムＤではアクセスできないかもしれない。リアルタイム・サーバ・アプリケーションは、ビデオがどこにあるかを認識し、割当てに適したシステムのリストを引き渡す。割当てファンクション（つまりシステム・リソース・マネージャ）は、次に後述するように、適格なシステムの中から最適なシステムを選択する。

【００５１】システムを選択するために用いられるロジックの実施例について、図9を参照して詳しく説明する。最初、システム・データ・テーブルを使用してシステム選択データ・エントリが作成される（ステップ９０

17

0）。1つの例として、システム選択データ配列は、割当てられるストリームに対応するため選択されるのに適した各システムについて、システム名、最後の間隔にシステムで開始されたストリームによって用いられる、計算済みサービス、及び利用可能サービス配列を含む。適格なシステムのリストは、割当てられる入力パラメータである。システムは、例えばCPU速度の順序で配列に置かれる。

【００５２】配列の構築に続いて、リアルタイム・データ・ストリームを配信するために充分なサービスが利用できるシステムを見つけるため、配列が検索される（ステップ９０２）。1つの例として、データ・ストリームに対応するため最小重要度で利用できる充分なサービスを持つシステムを見つけるまで、ロジックがシステム間で重要度を辿って配列を検索する。特に最初のチェックは、適格なシステムで未使用サービスを探すことである。これは重要度配列インデックス8である。ストリームに対応するのに充分な未使用サービスのあるシステムがない場合は、検索はシステムが見つかるまで、重要度を辿って進む。

【００５３】例えば、利用可能なサービスの各チェックの間に、システムで最後の間隔のとき開始されたストリームのサービス（システム選択データ配列内に格納されている）が、チェックされた重要度で利用できるサービスから引かれる。これは、他のストリームによって用いられるべきサービスがデータの中に現れる可能性が生じる前に、システムを選択しないでおくためである。

```
do imp = 8 to 1 by -1 while(system_selected="blank")
  do sys = 1 to number_of_eligible_systems
                    while (system_selected="blank")
    if serv_avail(imp,sys) -
                    started_stream_service(sys) >=
                        new_stream_service_rate
      set system_selected to name(sys)
  end
end
```

【００５８】図6に戻る。リクエストが実行されるシステムを選択した後、選択されたシステムの識別子（システム名等）、ステップ８００（図8）で計算される新しいストリームに必要なサービス、及び新しいレートが、マルチシステム環境の他のすべてのシステムに（従来のブロードキャストメカニズムにより）ブロードキャストされる（ステップ６０６）。特に各システムの割当てデータ受信ファンクションは、情報を受け取り、選択されたシステムのシステム・データ・テーブル・エントリにサービスを追加する（つまり図7の計算済みサービス７０６）。また、選択されたシステムは、新しいレートをプロセッサ利用状況データの現在の割当て済みレートに追加する。

【００５９】その後、選択済みシステム識別子は、リアルタイム・サーバ・アプリケーションに返される（ステップ６０８）。

18

*【００５４】充分なサービスを持つシステムが利用できない場合、割当てファンクションは失敗する（ステップ９０６）。ある実施例では、これは、充分なサービスが限度内で利用できる（つまり利用できるサービスが、2つ以上のシステムに分散している）場合でも真であるが、1つのシステムでは充分なサービスは利用できない。

【００５５】ただし、充分なサービスを持つシステムが利用できる場合は、割当てファンクションは先に進む（ステップ９０８）。

【００５６】システムの選択に用いられる疑似コードの1つの例を以下に示す。疑似コードでは、"name"はシステム名を、"started_stream_service"はこの間隔で開始されたストリームのサービスを、"serv_avail"は利用可能サービス配列を示す。

【００５７】システム選択疑似コード：

「変数」new_stream_service_rateは、割当てられる新しいストリームに対応するために必要なサービス・レート。new_stream_service_rateは割当てファンクションによって計算された。system_selectedは割当てファンクションの呼出し側に返されるシステム名。impは、システム選択データ配列の重要度レベル・データをループするループ・インデックス。sysは、システム選択データ配列のシステムごとのデータをループするループ・インデックス。

「初期化」system_selectedを"blank"にセットしてシステムが選択されていないことを示す。

【００６０】図3に戻る。割当てファンクションが実行された後、リアルタイム・サービング・アプリケーションを実行でき、これによりリアルタイム・データ・ストリームが配信される（ステップ３０６）。アプリケーションは割当てファンクションによって選択されたシステムで実行される。

【００６１】リアルタイム・アプリケーションが、リアルタイム・データ・ストリームを配信するために割当てられたプロセッサ・リソースを必要としなくなると、リソースは割当て解除ファンクションによって割当て解除される（ステップ３０８）。特にリアルタイム・サーバは、アプリケーションを実行するために選択されたシス

19

テムのシステム・リソース・マネージャの割当て解除ファンクションを起動する。これはリアルタイム・データ・ストリームが終了し、割当て済みプロセッサ・リソースを消費しなくなったことを示す。

【００６２】割当て解除ファンクションの実施例について、図１０を参照して説明する。１つの例として、プロセッサ・リソースを割当て解除するために、現在の割当て済みリアルタイム・レート５０６から入力データ・レートが引かれる（ステップ１０００）。従って現在のレートは、これらのリソースが他のデータ・ストリームの配信に利用できることを示す。その後、割当て解除は完了する。

【００６３】本発明に従って、リアルタイム・データ・ストリームの配信時に、クライアントは表示の一時停止を決定することができる。その場合、リアルタイム・サーバは、アプリケーションを実行するために選択されたシステムのシステム・リソース・マネージャの一時停止ファンクションを起動する。これはリアルタイム・データ・ストリームを表示するクライアントが、例えばビデオの表示を一時停止したことを示す。一時停止したストリームは、プロセッサの消費量には関係しないが、それらに必要なプロセッサ時間は保存される。従って、一時停止したストリームのデータ・レートは計算に用いられる。一時停止ファンクションの実施例については、図１１を参照して詳しく説明する。また再開ファンクションについても述べる。一時停止ファンクションと同様に、再開ファンクションも、アプリケーションを実行するために選択されたシステムで起動される。

【００６４】図１１を参照する。最初、クライアントがリアルタイム・データ・ストリームの配信を一時停止しているか、または一時停止の後に配信を再開しているかどうか確認される（照会１１００）。このオプションが一時停止なら、入力データ・レートが現在の一時停止リアルタイム・レート５１０に追加される（ステップ１１０２）。その後、本発明の一時停止ファンクションは完了する（ステップ１１０４）。

【００６５】他方、オプションが再開なら、現在の一時停止データ・レートから入力データ・レートが引かれ（ステップ１１０６）、再開ファンクションは完了する（ステップ１１０４）。

【００６６】本発明に従って、各MGDPC１５８は、そのシステムの性能データを計算し、またシステム・リソースの利用状況を監視して、監視結果をもとに特定の制御を調整する。例えばこれは、１０秒毎等、定義済み時間間隔で行われる。コントローラ１５８によって実行される操作の1つの例について、ここで図１２を参照して詳しく説明する。

【００６７】ある実施例で、MGDPCコントローラは、システムの性能データを計算する（ステップ１２００）。例えば、これはそのシステムの１８ウィンドウ枠

20

でのKB／秒レート当たりの最大サービス単位を計算する。この計算は、例えば１０秒に1回行われる。最大レートを確認する実施例を図１３に示す。以下これについて説明する。

【００６８】最初、ウィンドウ枠インデックス５１８がセットされ、ウィンドウ枠をスキャンするために最初のウィンドウ枠がポイントされ、高レートと呼ばれる変数が0に初期化される（ステップ１３００）。その後、KB／秒を配信するサービス・コスト（データ・レート）が次のように計算される。

【数2】リアルタイム・サービス・レート・ウィンドウ５１４÷（割当て済みリアルタイム・レート・ウィンドウ５１２－リアルタイム一時停止レート・ウィンドウ５１６）（ステップ１３０２）

【００６９】ステップ１３０２で計算された値は現在のサービス・レートであり、これも新しいレートと呼ばれる。新しいレートは、セーブされた高レートに対してチェックされる（照会１３０４）。割当て済みリアルタイム・レート・ウィンドウ枠５１２にセーブされた新しいレートが高レートより大きい場合、高レートは新しいレートにセットされる（ステップ１３０６）。新しい高レートをセーブした後、または現在のサービス・レートが高レートより小さいかまたは等しい場合は、さらにウィンドウ枠が他にあるかどうか確認される（照会１３０８）。特にウィンドウ枠インデックスが１８と比較され、１８より小さい場合は、この場合では、リアルタイム・ウィンドウ枠インデックス５１８が1つ増分される（ステップ１３１０）。制御は次にステップ１３０２”ウィンドウ枠のレート計算”に進む。

【００７０】他にウィンドウ枠がない場合は、セーブされた高レートがこのシステムのKB／秒レート当たりの最大サービス単位を表す。

【００７１】図１２に戻る。性能コントローラはまた、そのシステムのユーザ性能目標の達成を管理するため、ディスパッチ優先度等のシステム・リソース制御を調整する（ステップ１２０２）。

【００７２】システム・リソース制御の調整は、１つの例として、システムのアプリケーションの実行時にある間隔で実行される。ある実施例で、コントローラ１５８は１０秒毎に起動され、システム制御を調整する必要があるかどうか確認する。

【００７３】特にコントローラ１５８の制御調整機能は、性能目標の達成度を測定し、性能を改良する必要のあるユーザ性能目標クラスを選択し、関連付けられた作業単位のシステム制御パラメータ（つまりディスパッチ優先度等の被制御変数）を変更することによって選択されたユーザ性能目標クラスの性能を改良する。しかしディスパッチ優先度等の特定パラメータを変更する前に、性能コントローラ１５８は、本発明に従って、リアルタイム・サーバ・ビット１５２をチェックし、アプリケー

21

ションがリアルタイム・サーバ・アプリケーションかどうか確認する。ビットがオン、つまりリアルタイム・データ・ストリームの配信を示す場合は、クラスのディスパッチ優先度(優先度１５０）及び適応できる各作業単位のディスパッチ優先度(優先度１５４）は変わらないままである。これによりリアルタイム・データ・ストリームに一定量のリソースが保証される。

【００７４】リアルタイム・サーバ・ビットがオフのとき、ディスパッチ優先度は調整できる。非リアルタイム・アプリケーションに対するコントローラ１５８の処理については、C K Eilertらによる１９９５年2月3日付米国特許出願番号第０８／３８３０４２号、"Multi-System Resource Capping"を参照されたい。しかし本発明に従って、非リアルタイム・サービング作業単位に利用できるプロセッサ容量は、プロセッサ総容量から、リアルタイム・サービング作業単位に保証され、前記作業単位によって用いられるプロセッサ容量を引いたものである。

【００７５】前記のほかに、MGDPCコントローラは、キャッピング調整と呼ばれるタスクを実行する(ステップ１２０４）。キャッピングの調整も、先に述べた"Multi-System Resource Capping"で説明されているものと同様に働く。キャッピング調整ファンクションは、リソース消費量を、リソース・グループ最大値１３４で当該リソース・グループに指定された最大値に制限するため、各リソース・グループに割当てられた作業単位をディスパッチ不能(キャップ済み) にする時間を決定する。

【００７６】特に、時間の長さ(長さはプロセッサ速度に依存) は、例えば６４間隔等、いくつかの間隔に分けられ、キャッピング調整ファンクションは、各リソース・グループの作業単位にキャップを付ける間隔の個数を計算する。次に６４間隔のそれぞれに対応し、その間隔でリソース・グループにキャップを付けるかどうかを示すインジケータをセットする。これらのインジケータはキャップ・スライス・パターン１４０を形成する。キャップ・ビットは、"Multi-System Resource Capping"に述べられているように、６４個の間隔全体に比較的均一に分散される。

【００７７】６４の間隔のそれぞれでシステム・リソース・マネージャのキャップ・ビット操作ファンクションが起動され、キャップ・ビット・パターンがチェックされる。これにより、どのリソース・グループにキャップを付けるか確認され、作業単位制御ブロックにキャップ・ビットがセットされる。キャップ・ビットはディスパッチャ１２０によって問い合わせられ、作業単位をディスパッチできるかどうか確認される。本発明では、システム管理者が、リアルタイム・サービング・リソース・グループの最大値を指定し、キャッピング・ファンクションは、リアルタイム・サービング・アプリケーション

22

が、指定された最大プロセッサ容量以上を消費しないことを保証する。

【００７８】キャッピング・ファンクションの実施例について、図１４及び図１５を参照して詳しく説明する。図１４は、作業単位にキャップを付ける時間を、リソース・グループごとに確認するために用いられる制御の流れを示す。最初、プロセッサ消費最大値を有する処理すべきリソース・グループが他にあるかどうか確認される(ステップ１４００）。他にない場合、ファンクションは完了し、制御はステップ１４０２に移って終了する。他に処理すべきリソース・グループがある場合、リソース・グループがキャップ・スライスを現在セットしているか、または指定最大値より長いプロセッサ時間を現在消費しているかどうか確認される(照会１４０４）。

【００７９】前記のチェック結果が否定的なら、制御はステップ１４００に移り、次のリソース・グループが処理される。リソース・グループのキャップ・スライスが現在セットされているか、または指定最大値より長いプロセッサ時間をリソース・グループが現在消費している場合は、さまざまな初期化操作が実行される(ステップ１４０６）。例えばリソース・グループに割当てることのできる残りのプロセッサ・サービス単位を表す変数ＲＳＡが、リソース・グループに指定された最大値にセットされ、ローカル・プロセッサで消費されることが許可されたプロセッサ・サービス単位を表す変数ＬＳＡが0にセットされる。ローカル・プロセッサで許可されたプロセッサ・サービス単位は、ローカル及びすべてのリモートのプロセッサ上の作業の消費量と重要度をもとに配分される。配列変数ＬＳ及びＴＳもステップ１４０６で初期化される。ＬＳとＴＳは、重要度のレベルの個数に等しい数の要素を有する配列である。各重要度について、変数ＬＳ（ＩMP）が、最も最近の時間ウィンドウで、ローカル・システム上のその重要度で消費されたトータル・サービスにセットされる。これは、その重要度のサービス消費データをローカル・プロセッサのサービス消費ウィンドウの6 個の要素から加算することによって行われる(後述) 。各重要度で消費されたトータル・サービス、変数ＴＳ（ＩMP）も同様に計算される。

【００８０】その後、重要度変数（ＩMP）が最大重要度にセットされ、プロセッサ・サービス単位データを重要度によりループするために初期化される(ステップ１４０８）。次に、まだ配分できるプロセッサ・サービス単位があるかどうか確認される(照会１４１０）。プロセッサ・サービス単位がすべて配分されているなら、制御はステップ１４３０に移り、ローカル・プロセッサに配分されたプロセッサ・サービス単位数がキャップ・スライス数に変換される(後述) 。

【００８１】指定最大値を超えず、配分すべきプロセッサ・サービス単位が残っている場合、制御は照会１４１２に移り、ローカル、及びすべてのリモート・システム

23

上、現在の重要度I MPで、消費されたプロセッサ・サービス単位の総量ＴＳが、配分すべきプロセッサ・サービス単位より小さいかどうか確認される。現在の重要度ですべてのシステム上で消費されたプロセッサ・サービス単位の合計が、配分すべきプロセッサ・サービス単位より小さいかまたは等しい場合は、制御はステップ１４１４に移り、現在の重要度I MPで、ローカル・システムで消費されたプロセッサ・サービス単位数ＬＳが、許可されたローカル・プロセッサ・サービス単位の個数ＬＳＡに追加される。

【００８２】現在の重要度で消費されたプロセッサ・サービス単位の合計が、割当てるべきプロセッサ・サービス単位より大きい場合、制御はステップ１４１６に移り、許可された残りのプロセッサ・サービス単位の分数がローカル・システムに割当てられる。ローカル・システムに割当てられる残りのプロセッサ・サービス単位の分数は、ローカル・システムの現在の重要度で消費されたプロセッサ・サービス単位の数を、ローカル・システム及びすべてのリモート・システムの現在の重要度で消費されたプロセッサ・サービス単位の合計で割った数である。

【００８３】いずれの場合も、制御は次にステップ１４１８に移り、ローカル・システム及びすべてのリモート・システムで消費されたプロセッサ・サービス単位合計ＴＳ（ＩMP）が、割当てるべきプロセッサ・サービス単位数ＲＳＡから引かれる。次に、処理すべき重要度が他にあるかどうか確認される（照会１４２０）。他にある場合、制御はステップ１４２２に移り、現在の重要度変数、ＩMPが次に低い重要度にセットされる。その後、制御はステップ１４１０に戻り、先に述べたとおり、配分すべきプロセッサ・サービス単位があるかどうか確認される。

【００８４】しかし、重要度値がすべて処理されていることが確認されると（照会１４２０）、リソース・グループの消費量がその最大許容プロセッサ・サービス単位よりも少ないかどうか確認される（照会１４２４）。リソース・グループの消費量がその最大許容プロセッサ・サービス単位より少なくない場合、制御はステップ１４３０に移り、後述するように、リソース・グループのキャップ・スライスが計算される。

【００８５】他方、リソース・グループの消費量がその最大プロセッサ・サービス単位より少ない場合は、任意のシステム上のリソース・グループの任意の作業の最大重要度の作業がローカル・システムにあるかどうか確認される（照会１４２６）。追加されるプロセッサ・サービス単位は、リソース・グループの作業の重要度が最大値の作業を有するシステムに配分される。リソース・グループの作業の最大重要度の作業がローカル・システムにない場合、制御はステップ１４３０に移り、後述するように、リソース・グループのキャップ・スライスが計

24

算される。

【００８６】他方、リソース・グループの作業の最大重要度の作業がローカル・システムにある場合は、割当てることのできる残りのプロセッサ・サービス単位の分数がローカル・システムの作業に配分される（ステップ１４２８）。ローカル・システムの作業に割当てられた分数は、作業の重要度が最大のシステムの数で1を割った値である。

【００８７】ステップ１４３０で、許可されるプロセッサ・サービス単位の値（つまり最大値１３４）が、次のステップでキャップ・スライスの値に変換される。

１．許可される非キャップ・スライスの値を次のように計算する。

【数3】非キャップ・スライス＝ＬＳＡ＊（ＴＳＬＯＷ－ＣＳＬＯＷ）／ＬＳＥＲＯＷ

【００８８】ＬＳＡ　－　ローカル・プロセッサの許可サービス単位（つまりこのシステムの最大値）

ＴＳＬＯＷ　－　ウィンドウの総スライス数（例えば6＊６４＝３８４、定数）

ＣＳＬＯＷ　－　キャップ・スライス・ウィンドウでキャップが付いたスライスの値（後述）

ＬＳＥＲＯＷ　－　サービス・ウィンドウで消費されるローカル・プロセッサのサービス単位（後述）

キャップ・スライス・ウィンドウは、例えば6枠ウィンドウであり、例えばそれぞれ１０秒である。各ウィンドウ枠はその枠のキャップ付きスライスの値を保持する。サービス・ウィンドウも、例えば6枠ウィンドウで、各枠が１０秒で測定されたサービスを保持する。キャップ・スライス・ウィンドウとサービス・ウィンドウは、キャッピング・インデックスにより１から始めてインデックスが付けられる。増分は1で、６に達すると１に戻る。キャップ・スライス・ウィンドウ、サービス・ウィンドウ、及びキャッピング・インデックスは、プロセッサ利用状況データ１３８の一部である（しかしこれらは図5には示していない）。

２．非キャップ・スライスが計算して0になる場合は、これを1にセットする。時間の１００％にキャップを付けない。

３．キャップ・スライスを総スライス数（６４）から非キャップ・スライスを引いた値に等しくセットする。

４．キャップ・スライスの増加が総スライス数の2分の1より大きい場合は、キャップ・スライス数を総スライス数の2分の1だけ増やす。

５．キャップ・スライス数をリソース・グループ・テーブル・エントリ１３０に格納する。キャップ・スライス数は、現在のウィンドウ枠インデックスによってインデックスが付けられたキャップ・スライス配列要素に格納される。

【００８９】キャップ・スライスに許可されたプロセッサ・サービス単位数の変換に続いて、処理はステップ1

25

４１０に戻る。

【００９０】キャップを付けるべきキャップ・スライス数を求める他に、各リソース・グループのキャップ・スライスが、キャップ・ビット・マニピュレータが問い合わせをするため、キャップ・スライス・パターンに比較的均一に分散される。キャップ付きスライス数が最大のリソース・グループには特定のスライスが最初に割当てられ、次に最大のキャップ・スライス数を要するリソース・グループが次に処理され、というように、キャップ付きスライスを有するすべてのリソース・グループが処理されていく。システム全体で、同じ個数のリソース・グループ±1グループに、キャップ・スライス・パターンの６４個のスライスのそれぞれの間で、特定スライスが特定のリソース・グループに割当てられるとき、各スライスの間でキャップが付けられるリソース・グループの値のカウントを維持することによってキャップが付けられる。このカウントはテストされ、グループに特定スライスを割当てるかどうか確認される。６４個のすべてのスライスが割当てられるまでは、２つのリソース・グループにスライスが割当てられることはない。６４個のすべてのスライスが2回割当てられるまでは、３つのグループにスライスが割当てられることはない。以下同様である。

【００９１】キャップ・スライスを分散させる実施例について、図１５を参照して詳しく説明する。最初、配列要素を0にセットすることによってシステム・データ、キャップ・スライス・カウント配列（ＳＣＡ）が初期化される（ステップ１５００）。スライス・カウント配列はカウントの配列である。各エントリはそのスライスの間にキャップが付けられるグループの値を保持する。キャップ・スライス・カウント配列には６４のエントリがある。各エントリは時間の６４分の1を表す。

【００９２】その後、まだ処理されていないキャップ・スライスの値が最大のリソース・グループが選択される（ステップ１５０２）。処理すべきリソース・グループがない場合は、制御は終了する（ステップ１５０４）。処理すべきリソース・グループが他にある場合は制御はステップ１５０６に進む。

【００９３】ステップ１５０６でリソース・グループ・データが初期化される。特にスライス増分（ＳＩ）、スライス・カウント基準（ＳＣＣ）、及びスライス・カウント配列インデックス（ＳＣＡＩ）、が次のように初期化される。（ＳＩ－1）は、リソース・グループの各キャップ・スライスの間でスキップできるスライス数であり、リソース・グループのキャップ・スライスは６４個の可能なキャップ・スライスで比較的均一に分散される。ＳＩは、グループのキャップ・スライス数で分けられた６４にセットされる。ＳＣＣはスライス・カウント配列の最大値にセットされる。スライス・カウント配列のすべてのカウントが同じ値なら、ＳＣＣに1が追加される。ＳＣＡＩはキャップ・スライス・パターンの最初のスライスで始めるため、１にセットされる。

26

【００９４】次に、ＳＣＡ（ＳＣＡＩ）セルのカウントが、このグループの基準（ＳＣＣ）に対してチェックされる（照会１５０８）。スライスがＳＣＣ個のグループに割当てられているかがチェックされ、またはすでにこのグループに割当てられている場合は、制御はステップ１５１０に移り、ここでＳＣＡＩに1が追加される。ＳＣＡＩが６５に増分されると、これは1にリセットされる（つまりＳＣＡＩは６４を法とする）。スライス・カウント配列のすべてのカウントが同じ値であれば、ＳＣＣに1が追加される。制御は次にステップ１５０８に戻る。

【００９５】スライスがＳＣＣより小さい個数のリソース・グループに割当てられていて、処理されている現在のグループに割当てられていない場合、制御はステップ１５１２に移り、そこでスライスがグループに割当てられ、ＳＣＡのスライスの利用状況カウントに1が追加される。スライスは、リソース・グループのキャップ・スライス・パターン１４０のＳＣＡＩに対応したビットをセットすることによってリソース・グループに割当てられる。制御は次に照会１５１４に移る。

【００９６】照会１５１４では、現在のリソース・グループに割当てるスライスが他にあるかどうか確認される。特定のスライスがすべてのグループのキャップ・スライスに割当てられている場合、制御はステップ１５０２に戻り、他のリソース・グループがチェックされる。しかしグループに他の特定のスライスを割当てる必要がある場合は、制御はステップ１５１６に移る。

【００９７】ステップ１５１６では、ＳＣＡＩにＳＩが追加され、次にチェックすべき特定のスライスがセットされる。ＳＣＡＩが６４より大きい場合はＳＣＡＩから６４が引かれる。ＳＣＡのすべてのカウントが同じ値なら、ＳＣＣに1が追加される。次に制御はステップ１５０８に戻る。その後、キャップ・スライスの分散が完了する。

【００９８】図１２に戻る。コントローラ１５８は、調整を実行するほかに、システム・データを、例えば１０秒毎に送る（ステップ１２０６）。例えば、コントローラ１５８は、各重要度で利用できるサービスと未使用サービスと共に、このシステム（図１３等）で計算された最大サービス・レートをシステム・データ・テーブルに送る。またＣＰＵ速度もシステム・データ・テーブルに送る。

【００９９】上に述べたのは、選択された量のリソースをリアルタイム・サーバに提供し、また、システムの他のタイプのアプリケーション及び他の作業を処理するため、選択された量のリソースが予約されることを保証する、プロセッサ・リソース管理機能である。本発明の管理機能は、リクエストに対応でき、最小優先度の作業を

置き換えることのできるシステムをマルチシステム環境から選択する。

【０１００】上に述べたコンピュータ装置は1つの例にすぎない。他のシステムや環境も、本発明の主旨から逸脱することなく本発明を取り入れ、あるいは利用することができる。

【０１０１】本発明は、例えばコンピュータ使用可能媒体を有する製品（コンピュータのプログラム・プロダクト等）に加えることができる。媒体には、本発明の機能を提供し促進するため、コンピュータ読取り可能プログラム・コード手段が具体化される。製品はコンピュータ装置の一部として追加するか、または個別に販売することもできる。

【０１０２】ここに示したフローチャートも１つの例にすぎない。これらの図やステップ（あるいは操作）は本発明の主旨から逸脱することなく変更することができる。例えばステップは異なる順序で実行できる。あるいはステップは追加、削除、または変更可能である。これらの変形はすべて特許請求範囲とみなされる。

【０１０３】以上、好適な実施例が詳細に図示及び説明されたが、様々な変更、追加、代替等が、本発明の趣旨から逸脱することなく実現可能であり、従って本発明の範囲内で考慮されることが当業者には理解されよう。

【０１０４】まとめとして、本発明の構成に関して以下の事項を開示する。

【０１０５】（１）複数の装置で構成されたマルチシステム環境のプロセッサ・リソースを管理する方法であって、少なくともリアルタイム・アプリケーションを含むリアルタイム・アプリケーションのグループに選択された限度を超えないある量のプロセッサ・リソースを前記マルチシステム環境のリアルタイム・アプリケーションに割当て、選択された量の前記プロセッサ・リソースは、前記マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できるステップと、前記複数の装置のうち、選択された装置で前記リアルタイム・アプリケーションを処理するステップと、を含む、方法。

（２）前記装置を選択するステップを含み、前記選択ステップは、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するある量の前記プロセッサ・リソースが、選択された装置の他の作業に与える影響は最小である装置を複数の装置のうち少なくとも２つの装置から選択するステップを含む、前記（１）記載の方法。

（３）前記割当てステップは、少なくとも２つの装置から、リアルタイム・データ・レートを配信するための、所与の時間内での、最大サービス・コストを求めるステップと、前記リアルタイム・データ・ストリームを配信するサービス・レートを、前記最大サービス・コストを使用して計算するステップと、前記サービス・レートが

前記限度内にあるかどうか確認するステップであって、前記選択ステップは、前記サービス・レートが前記限度内にあるときに実行される、前記確認するステップと、を含む、前記（２）記載の方法。

（４）前記確認ステップは、少なくとも２つの装置の高サービス・コストを確認するステップと、前記最大サービス・コストを取得するため、前記少なくとも２つの高サービス・コストを比較するステップと、を含む、前記（３）記載の方法。

（５）少なくとも選択されていない任意の装置に、選択された装置について通知するステップを含む、前記（２）記載の方法。

（６）前記通知ステップは、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するために計算されたサービス・レートの指示を、前記少なくとも選択されていない任意の装置に送るステップを含む、前記（５）記載の方法。

（７）前記処理ステップは、リアルタイム・アプリケーションの前記グループに選択された前記プロセッサ・リソースの最大量を、前記リアルタイム・アプリケーションが超えるのを防ぐステップを含む、前記（１）記載の方法。

（８）前記防止ステップは、前記リアルタイム・アプリケーションの作業単位が前記最大量を超えずにはディスパッチ不能であることを示すステップを含む、前記（７）記載の方法。

（９）前記リアルタイム・アプリケーションの処理が求められていることを、前記複数の装置のいくつかに示すステップを含む、前記（１）記載の方法。

（１０）前記指示ステップは、前記リアルタイム・アプリケーションに選択されたディスパッチ優先度を前記いくつかの装置内に設定するステップと、前記ディスパッチ優先度は調整されないことを、前記いくつかの装置内に指定するステップと、を含む、前記（９）記載の方法。

（１１）前記複数の装置が、前記限度を利用できるようにするステップを含む、前記（１）記載の方法。

（１２）複数の装置で構成されたマルチシステム環境のプロセッサ・リソースを管理するために、コンピュータ読取り可能プログラム・コード手段が具体化されたコンピュータ使用可能媒体を含む製品であって、前記コンピュータ読取り可能プログラム・コード手段は、少なくともリアルタイム・アプリケーションを含むリアルタイム・アプリケーションのグループに選択された限度を超えない、ある量のプロセッサ・リソースを前記マルチシステム環境のリアルタイム・アプリケーションに割当てることを、コンピュータによって可能にし、選択された量の前記プロセッサ・リソースは前記マルチシステム環境の少なくとも１つの非リアルタイム・アプリケーションに引き続き利用できる、コンピュータ読取り可能プログ

29

ラム・コード手段と、前記複数の装置のうち選択された装置での前記リアルタイム・アプリケーションの処理をコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、を含む、製品。

（１３）前記装置の選択をコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含み、コンピュータによって選択を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するある量の前記プロセッサ・リソースが、選択された装置の他の作業に与える影響は最小である装置を、複数の装置のうち少なくとも２つの装置から選択することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１２）記載の製品。

（１４）コンピュータによって割当てを可能にする、前記コンピュータ読取り可能プログラム・コード手段は、リアルタイム・データ・レートを配信するため、所与の時間内の最大サービス・コストを少なくとも２つの装置から求めることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、前記リアルタイム・データ・ストリームを配信するサービス・レートを、前記最大サービス・コストを使用して計算することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、前記サービス・レートが前記限度内にあるかどうか確認することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段であって、前記選択ステップは前記サービス・レートが前記限度内にあるとき実行される、前記コンピュータ読取り可能プログラム・コード手段と、を含む、前記（１３）記載の製品。

（１５）コンピュータによって確認を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、少なくとも２つの装置の高サービス・コストを求めることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、前記最大サービス・コストを取得するため、前記少なくとも２つの高サービス・コストを比較することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、を含む、前記（１４）記載の製品。

（１６）少なくとも選択されていない任意の装置に、選択された装置について通知することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１３）記載の製品。

（１７）コンピュータによって通知を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションのリアルタイム・データ・ストリームを配信するために計算されたサービス・レートの指示を、少なくとも選択されていない任意の装置に送ることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１６）記載の製品。

30

（１８）コンピュータによって処理を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、リアルタイム・アプリケーションの前記グループに選択された前記プロセッサ・リソースの最大量を前記リアルタイム・アプリケーションが超えるのを防ぐことをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１２）記載の製品。

（１９）コンピュータによって防止を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションの作業単位が前記最大量を超えずにはディスパッチ不能であることを指示することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１８）記載の製品。

（２０）前記リアルタイム・アプリケーションの処理が求められることを前記複数の装置のうちいくつかの装置に示すことをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１２）記載の製品。

（２１）コンピュータによって指示を可能にする、前記コンピュータ読取り可能プログラム・コード手段は、前記リアルタイム・アプリケーションに選択されたディスパッチ優先度を前記いくつかの装置内に設定することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、前記ディスパッチ優先度は調整されないことを前記いくつかの装置内に指定することをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段と、を含む、前記（２０）記載の製品。

（２２）前記複数の装置が前記限度を利用できるようにすることをコンピュータによって可能にする、コンピュータ読取り可能プログラム・コード手段を含む、前記（１２）記載の製品。

【図面の簡単な説明】

【図1】本発明の管理機能を取り入れて使用するマルチシステム環境の1つの例を示す図である。

【図2】本発明に従って用いられるディスパッチ優先度の1つの例を示す図である。

【図3】本発明に従って、リアルタイム・アプリケーションの処理に関連して用いられるロジックの実施例を示す図である。

【図4】本発明に従って、図3の識別機能に関連付けられるロジックの1つの例を示す図である。

【図5】本発明に従って用いられるプロセッサ利用状況データの1つの例を示す図である。

【図6】本発明に従って、リアルタイム・アプリケーションにプロセッサ・リソースを割当てるために用いられ

るロジックの実施例を示す図である。

【図7】本発明に従って用いられるシステム・データ・テーブル・エントリの1つの例を示す図である。

【図8】本発明に従って、リアルタイム・データ・ストリームを配信するため、システムに充分な容量があるかどうかチェックするために用いられるロジックの実施例を示す図である。

【図9】本発明に従って、リアルタイム・データ・ストリームを配信するため、システムを選択するために用いられるロジックの実施例を示す図である。

【図１０】本発明に従って、リアルタイム・アプリケーションに割当てられるリソースの割当て解除時に用いられるロジックの実施例を示す図である。

【図１１】本発明に従って、リアルタイム・アプリケーションの実行の一時停止／再開時に用いられるロジックの実施例を示す図である。

【図１２】本発明のMGDPC（multisystem goal driven performance controller）のさまざまな操作の1つの例を示す図である。

【図１３】本発明に従って、あるシステムの高サービス単位レートを計算するために用いられるロジックの実施例を示す図である。

【図１４】本発明に従って、図１２のMGDPCの操作のキャッピング・ファンクション実行時に用いられるロジックの実施例を示す図である。

【図１５】本発明に従って、図１２のMGDPCの操作のキャッピング・ファンクション実行時に用いられるロジックの実施例を示す図である。

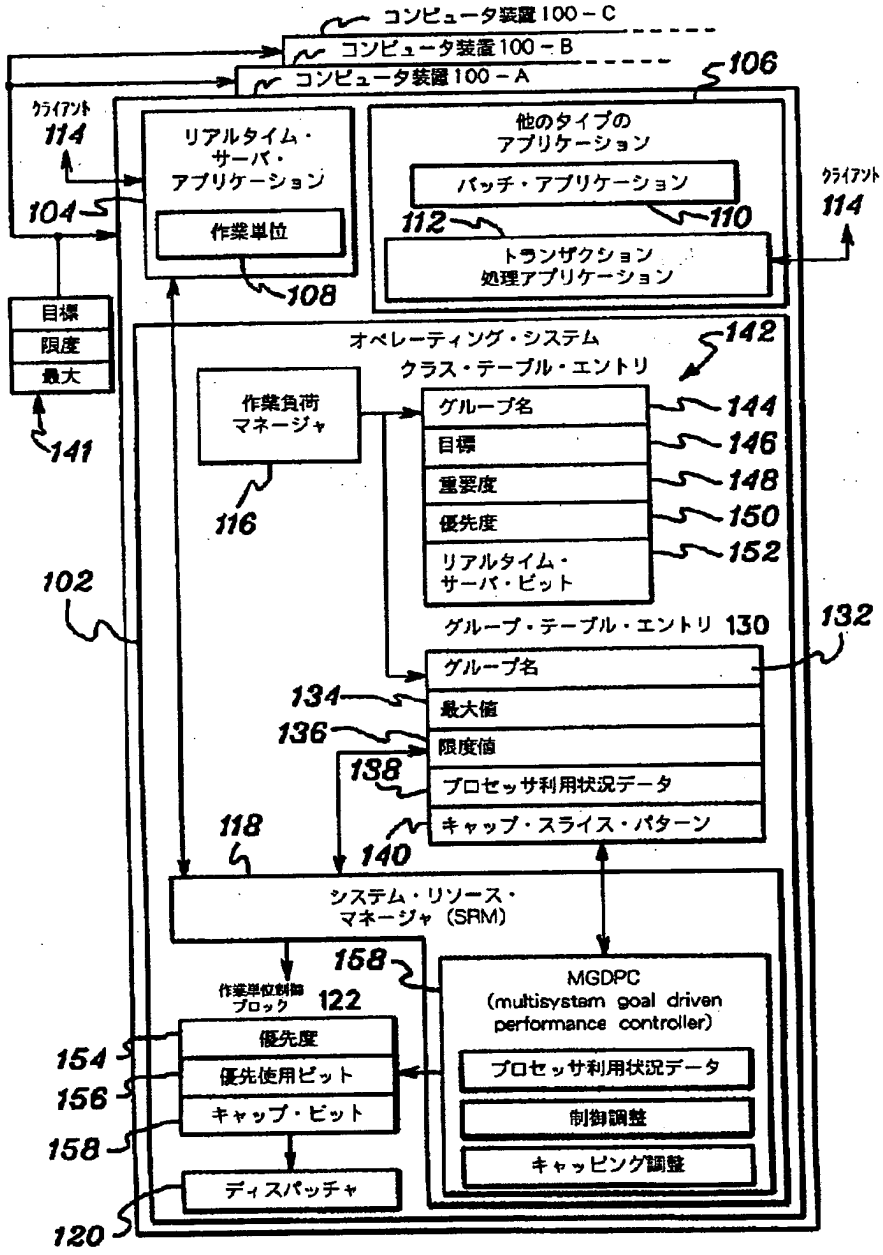
【図１６】本発明に従って、図１２のMGDPCの操作のキャッピング・ファンクション実行時に用いられるロジックの実施例を示す図である。

【符号の説明】

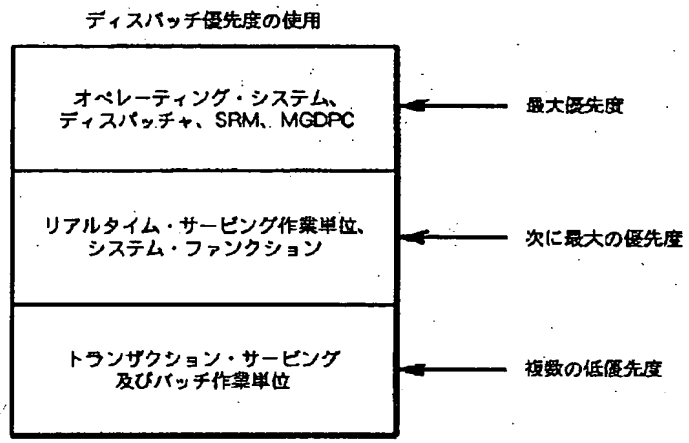
１００－A、１００－B、１００－C　コンピュータ装置
１０２　オペレーティング・システム
１０４　リアルタイム・サーバ・アプリケーション
１０６　他のタイプのアプリケーション
１０８　作業単位

１１０　バッチ・アプリケーション
１１２　トランザクション処理アプリケーション
１１４　サービング・クライアント
１１６　作業負荷マネージャ
１１８　システム・リソース・マネージャ
１２０　ディスパッチャ
１２２　作業単位制御ブロック
１３０　リソース・グループ・テーブル・エントリ
１３２、１４４　リソース・グループ名
１３４　プロセッサ消費最大値
１３６　プロセッサ消費限度値
１３８　プロセッサ利用状況データ
１４０　キャップ・スライス・パターン
１４１　外部記憶装置
１４２　クラス・テーブル・エントリ
１４６　ユーザ性能目標
１４８　ユーザ性能目標の相対重要度
１５０　ディスパッチ優先度
１５２　リアルタイム・サーバ・ビット
１５４　ディスパッチ優先度フィールド
１５６　優先使用ビット・フィールド
１５８　キャップ・ビット・フィールド
５０２　グループ・サービス・レート
５０４　前の割当て済みリアルタイム・レート
５０６　現在の割当て済みリアルタイム・レート
５０８　前の一時停止リアルタイム・レート
５１０　現在の一時停止リアルタイム・レート
５１２　割当て済みリアルタイム・レート・ウィンドウ
５１４　リアルタイム・サービス・レート・ウィンドウ
５１６　リアルタイム一時停止レート・ウィンドウ
５１８　リアルタイム・ウィンドウ枠インデックス
７００　エントリ
７０２　システム名
７０４　ウィンドウの高サービス単位／KBレート
７０６　計算済みサービス
７０８　利用可能サービス配列
７１０　単一エンジンCPU速度

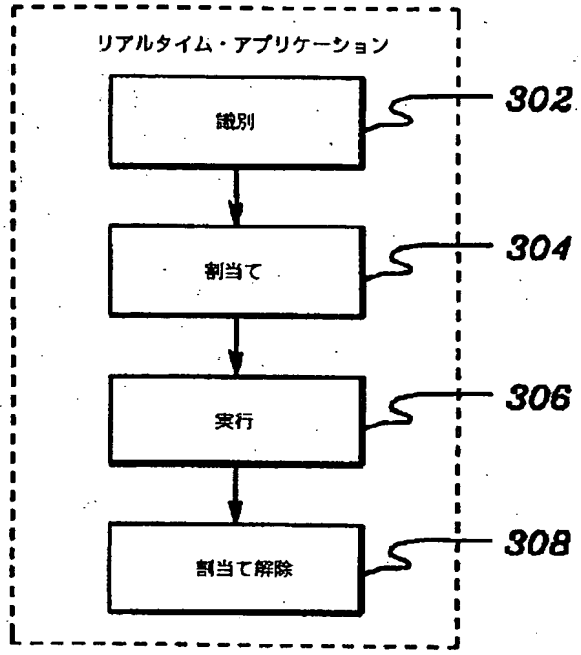
【 図1 】


コンピュータ装置100－C
コンピュータ装置100－B
コンピュータ装置100－A
106
クライアント
114
リアルタイム・
サーバ・
アプリケーション
作業単位
104
108
他のタイプの
アプリケーション
バッチ・アプリケーション
112
110
トランザクション
処理アプリケーション
クライアント
114
目標
限度
最大
141
オペレーティング・システム
クラス・テーブル・エントリ
142
作業負荷
マネージャ
116
グループ名
目標
重要度
優先度
リアルタイム・
サーバ・ビット
144
146
148
150
152
102
グループ・テーブル・エントリ 130
132
グループ名
最大値
限度値
プロセッサ利用状況データ
キャップ・スライス・パターン
134
136
138
140
118
システム・リソース・
マネージャ（SRM）
158
MGDPC
(multisystem goal driven
performance controller)
プロセッサ利用状況データ
制御調整
キャッピング調整
作業単位制御
ブロック
122
優先度
優先使用ビット
キャップ・ビット
154
156
158
ディスパッチャ
120

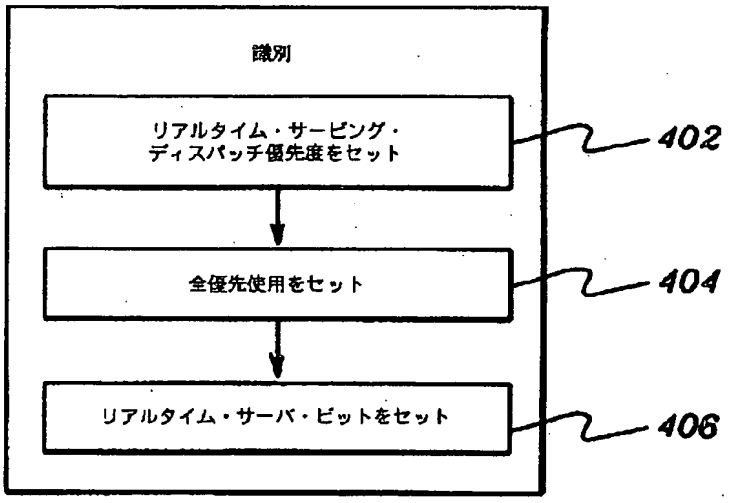
【 図2 】


ディスパッチ優先度の使用
オペレーティング・システム、
ディスパッチャ、SRM、MGDPC
最大優先度
リアルタイム・サービング作業単位、
システム・ファンクション
次に最大の優先度
トランザクション・サービング
及びバッチ作業単位
複数の低優先度

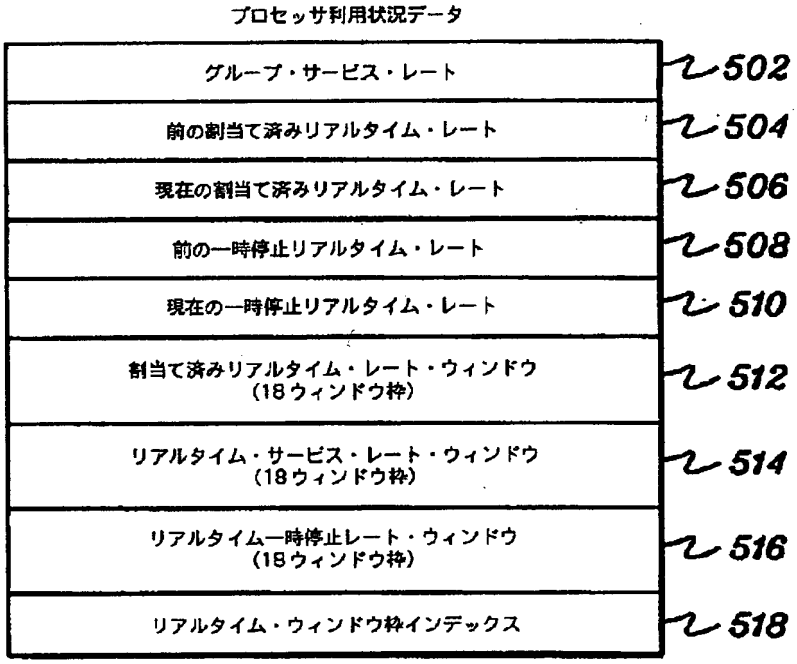
【 図3 】


リアルタイム・アプリケーション
識別
302
割当て
304
実行
306
割当て解除
308

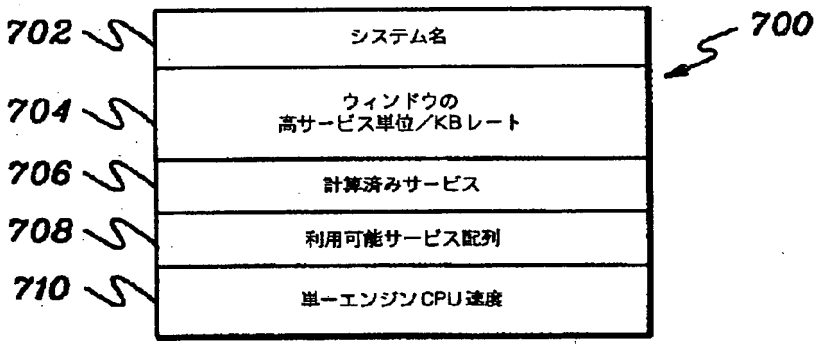
【 図4 】


識別
リアルタイム・サービング・
ディスパッチ優先度をセット
402
全優先使用をセット
404
リアルタイム・サーバ・ビットをセット
406

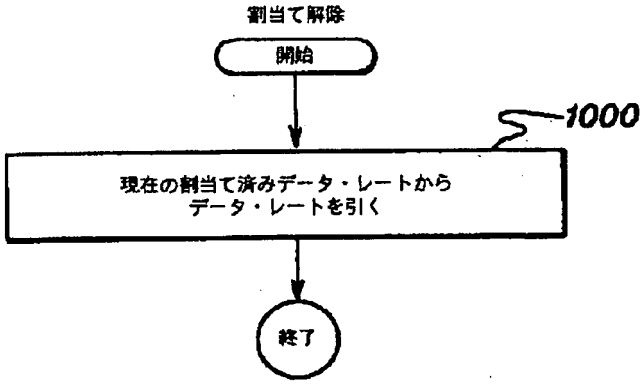
【 図5 】


プロセッサ利用状況データ
グループ・サービス・レート
502
前の割当て済みリアルタイム・レート
504
現在の割当て済みリアルタイム・レート
506
前の一時停止リアルタイム・レート
508
現在の一時停止リアルタイム・レート
510
割当て済みリアルタイム・レート・ウィンドウ
(18ウィンドウ枠)
512
リアルタイム・サービス・レート・ウィンドウ
(18ウィンドウ枠)
514
リアルタイム一時停止レート・ウィンドウ
(18ウィンドウ枠)
516
リアルタイム・ウィンドウ枠インデックス
518

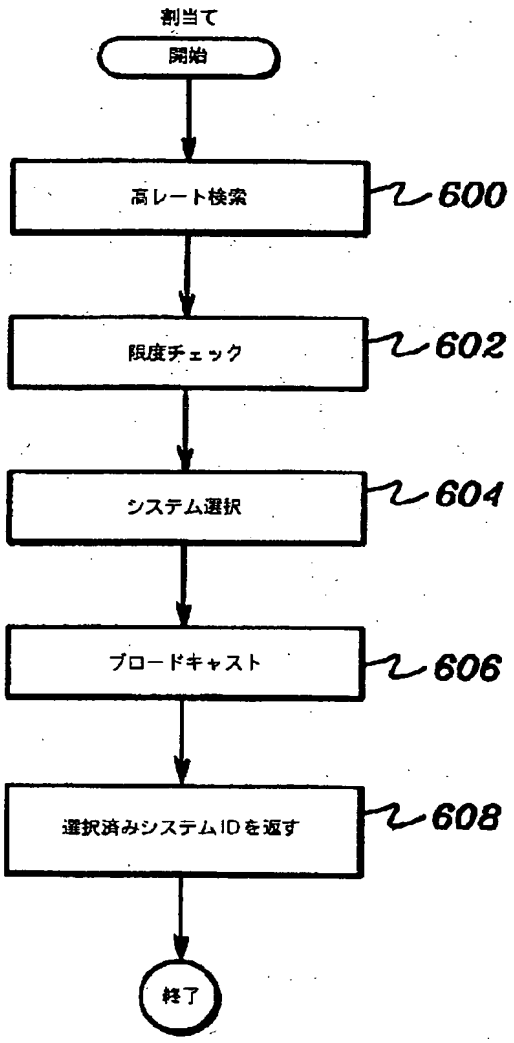
【 図7 】


700
702
システム名
704
ウィンドウの
高サービス単位／KBレート
706
計算済みサービス
708
利用可能サービス配列
710
単一エンジンCPU速度

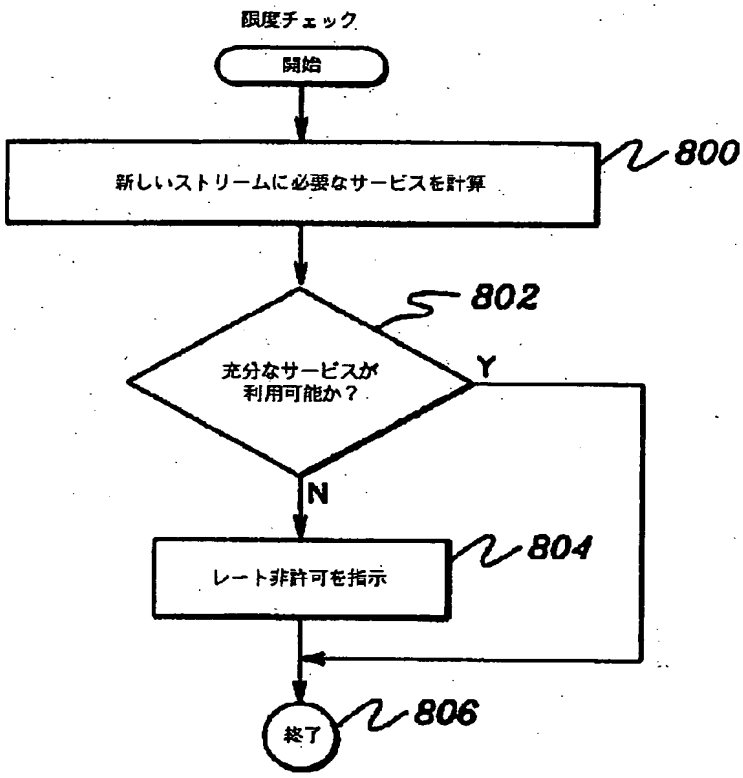
【 図１０ 】


割当て解除
開始
1000
現在の割当て済みデータ・レートから
データ・レートを引く
終了

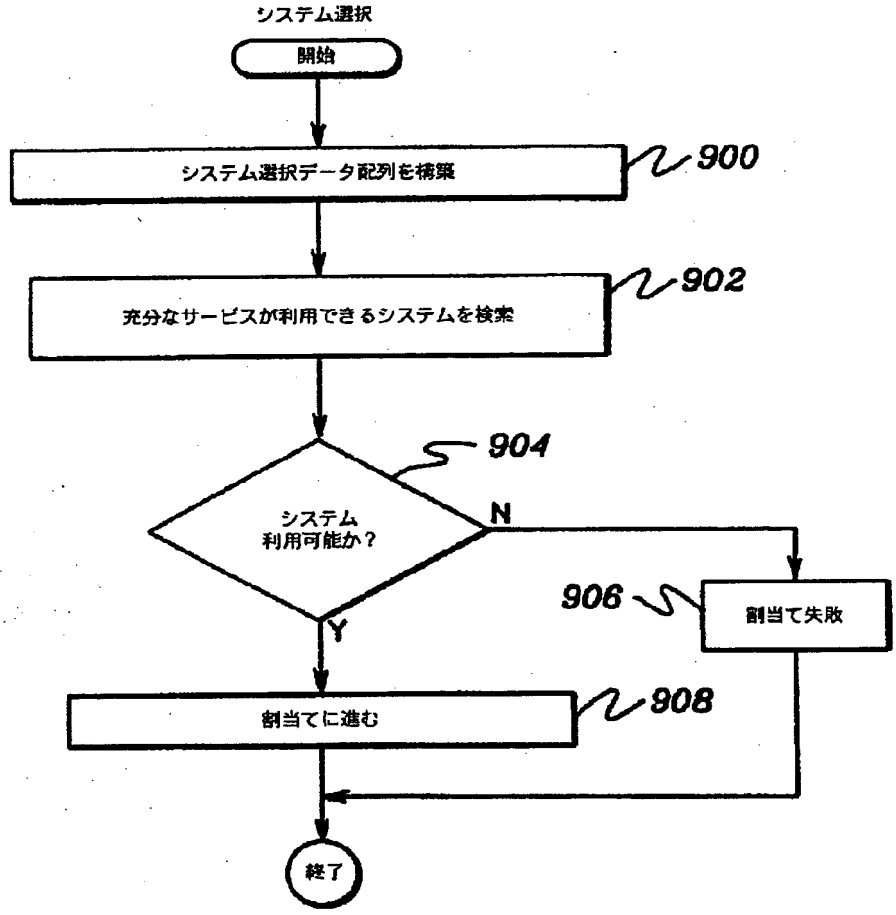
【図６】


割当て
開始
高レート検索
600
限度チェック
602
システム選択
604
ブロードキャスト
606
選択済みシステムIDを返す
608
終了

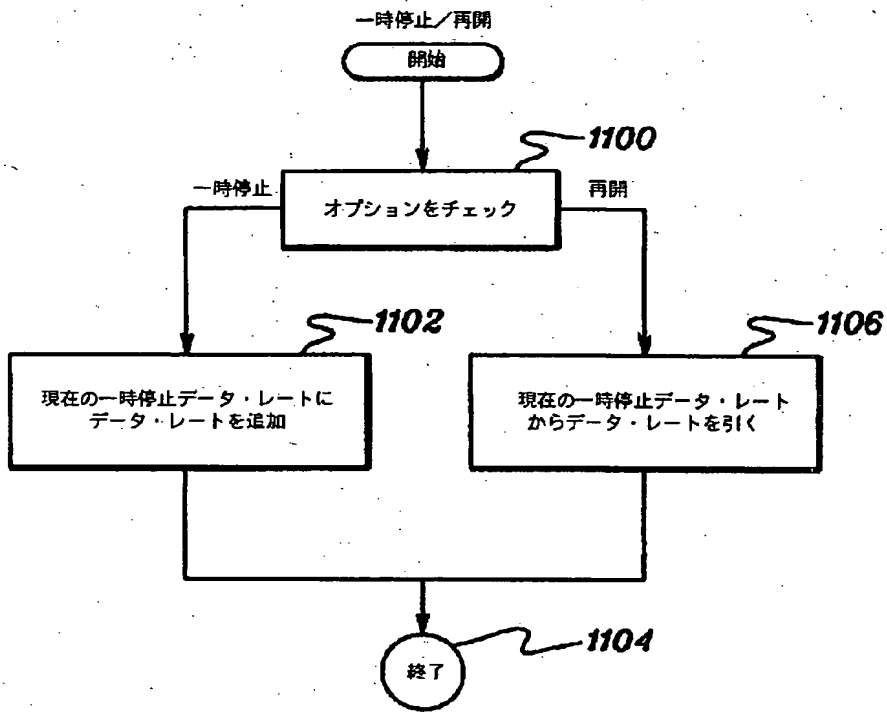
【図８】


限度チェック
開始
新しいストリームに必要なサービスを計算
800
802
充分なサービスが
利用可能か？
Y
N
レート非許可を指示
804
終了
806

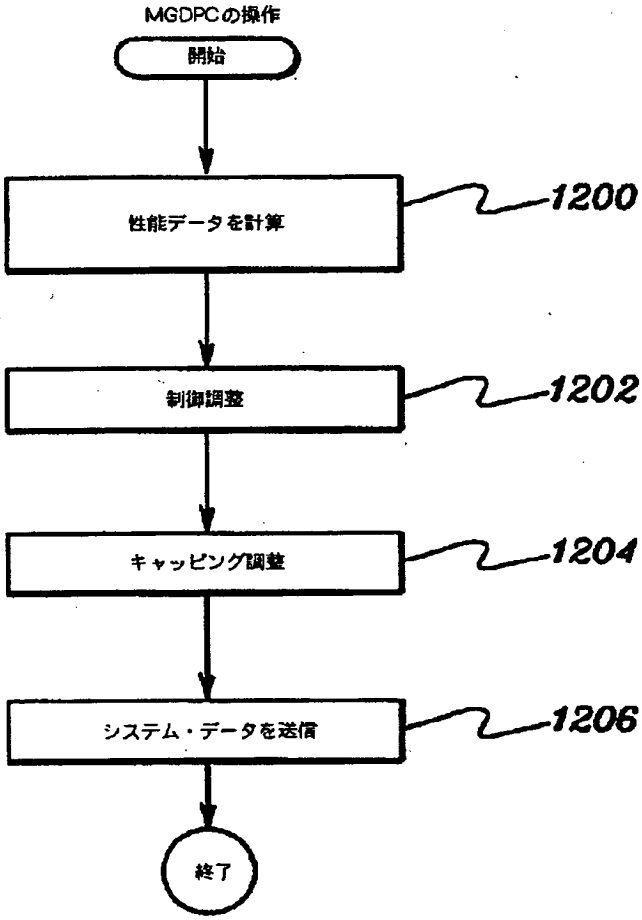
【 図9 】


システム選択
開始
システム選択データ配列を構築
900
充分なサービスが利用できるシステムを検索
902
904
システム
利用可能か？
N
Y
906
割当て失敗
割当てに進む
908
終了

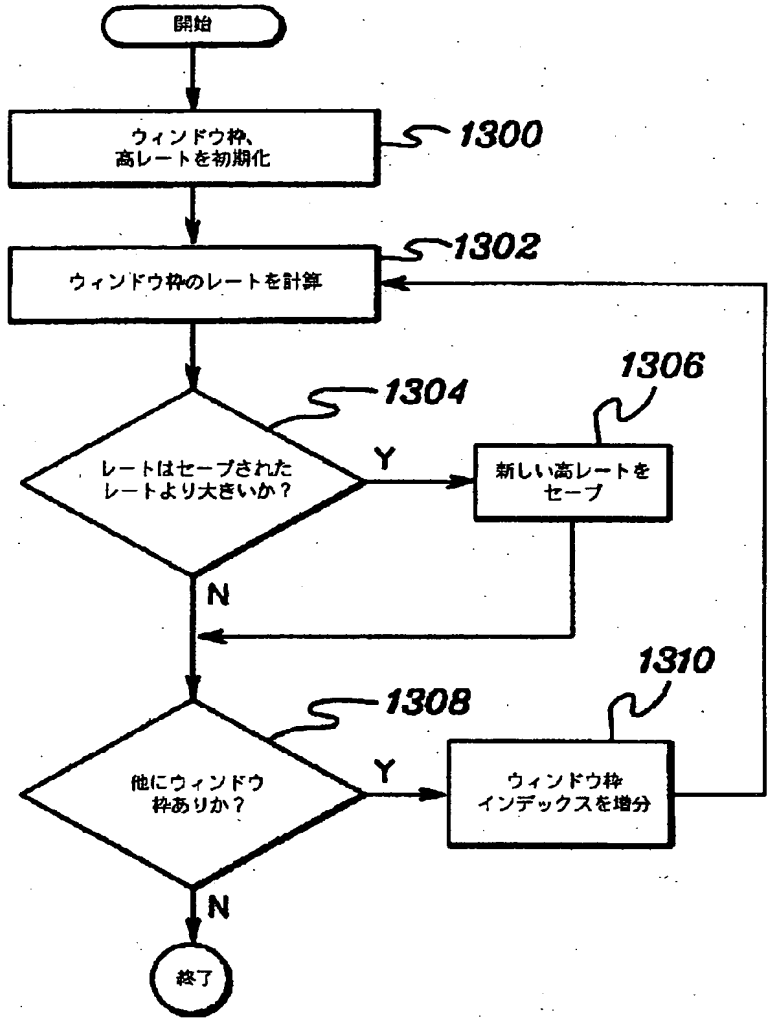
【図１１】


一時停止／再開
開始
1100
一時停止
オプションをチェック
再開
1102
1106
現在の一時停止データ・レートに
データ・レートを追加
現在の一時停止データ・レート
からデータ・レートを引く
1104
終了

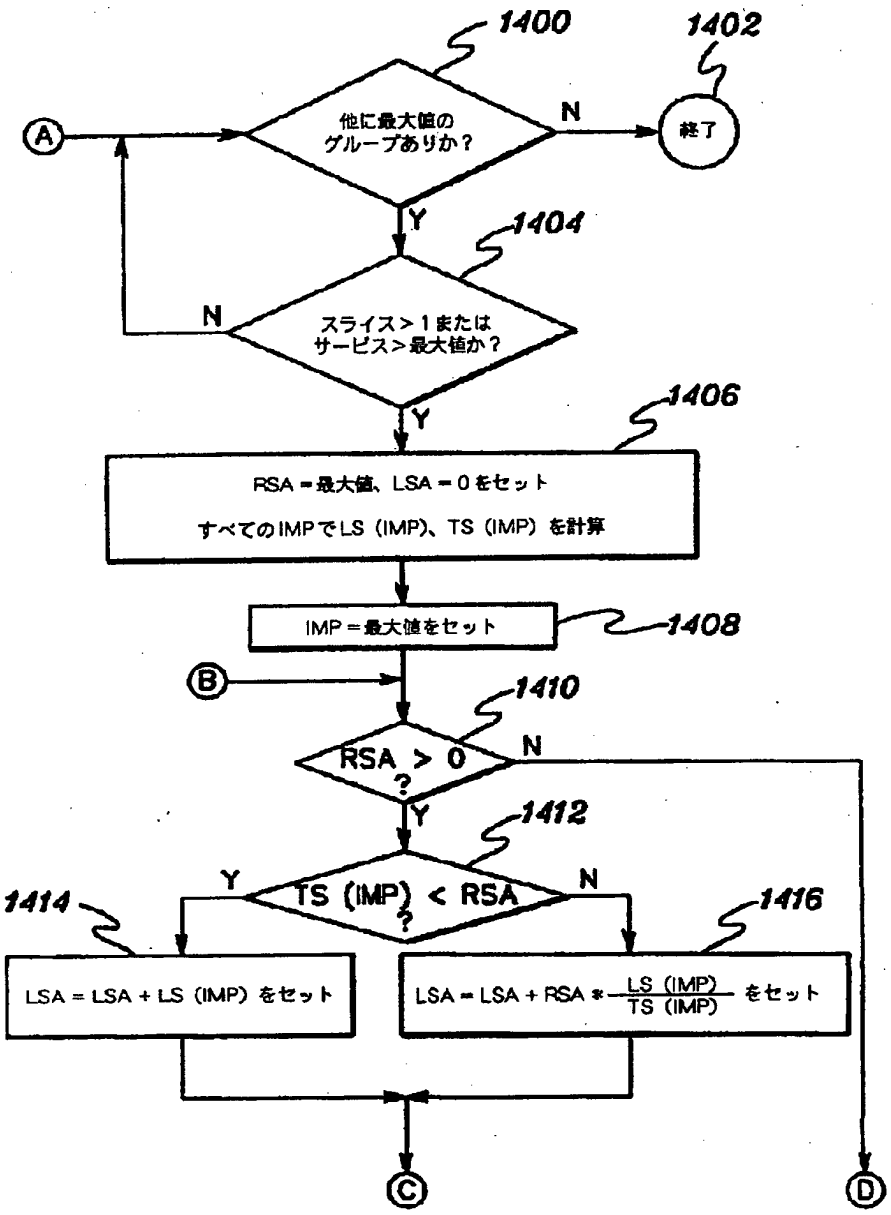
【 図１２ 】


MGDPCの操作
開始
性能データを計算
1200
制御調整
1202
キャッピング調整
1204
システム・データを送信
1206
終了

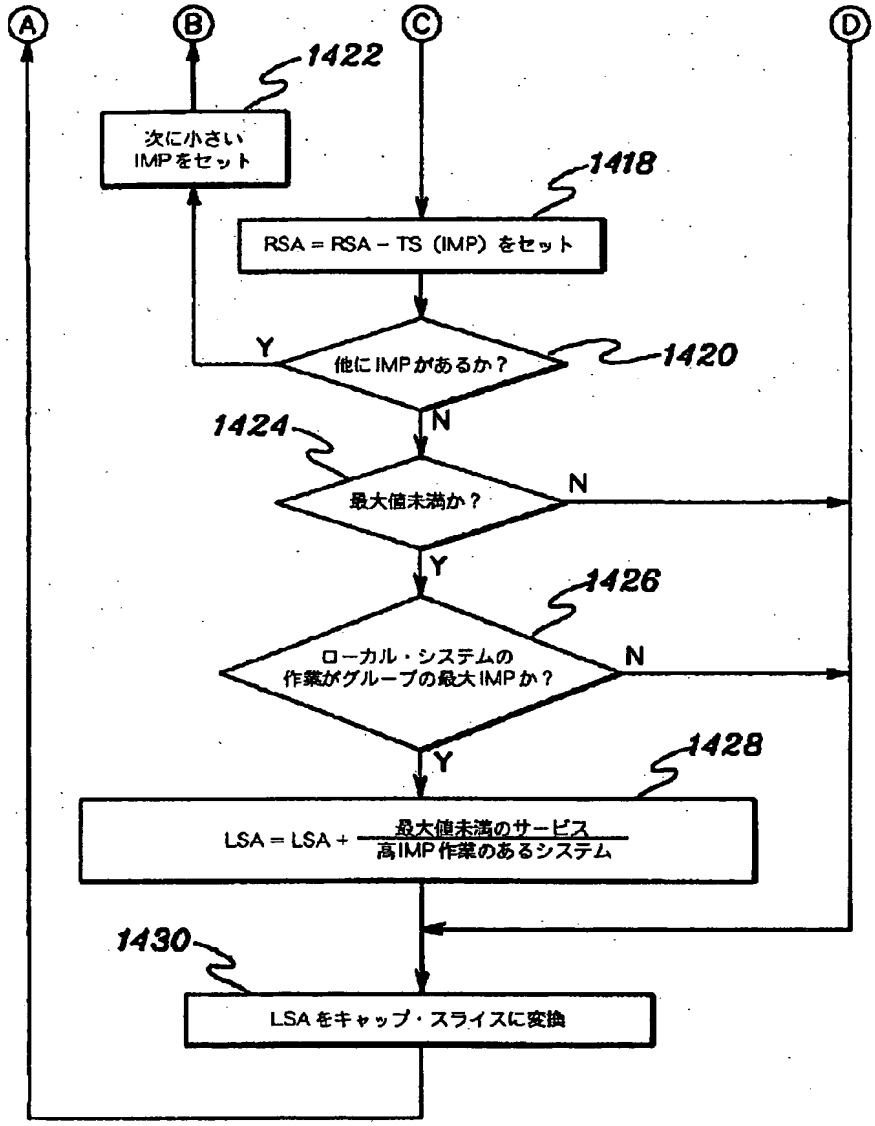
【図１３】


開始
ウィンドウ枠、
高レートを初期化
1300
ウィンドウ枠のレートを計算
1302
1304
レートはセーブされた
レートより大きいか？
Y
1306
新しい高レートを
セーブ
N
1308
他にウィンドウ
枠ありか？
Y
1310
ウィンドウ枠
インデックスを増分
N
終了

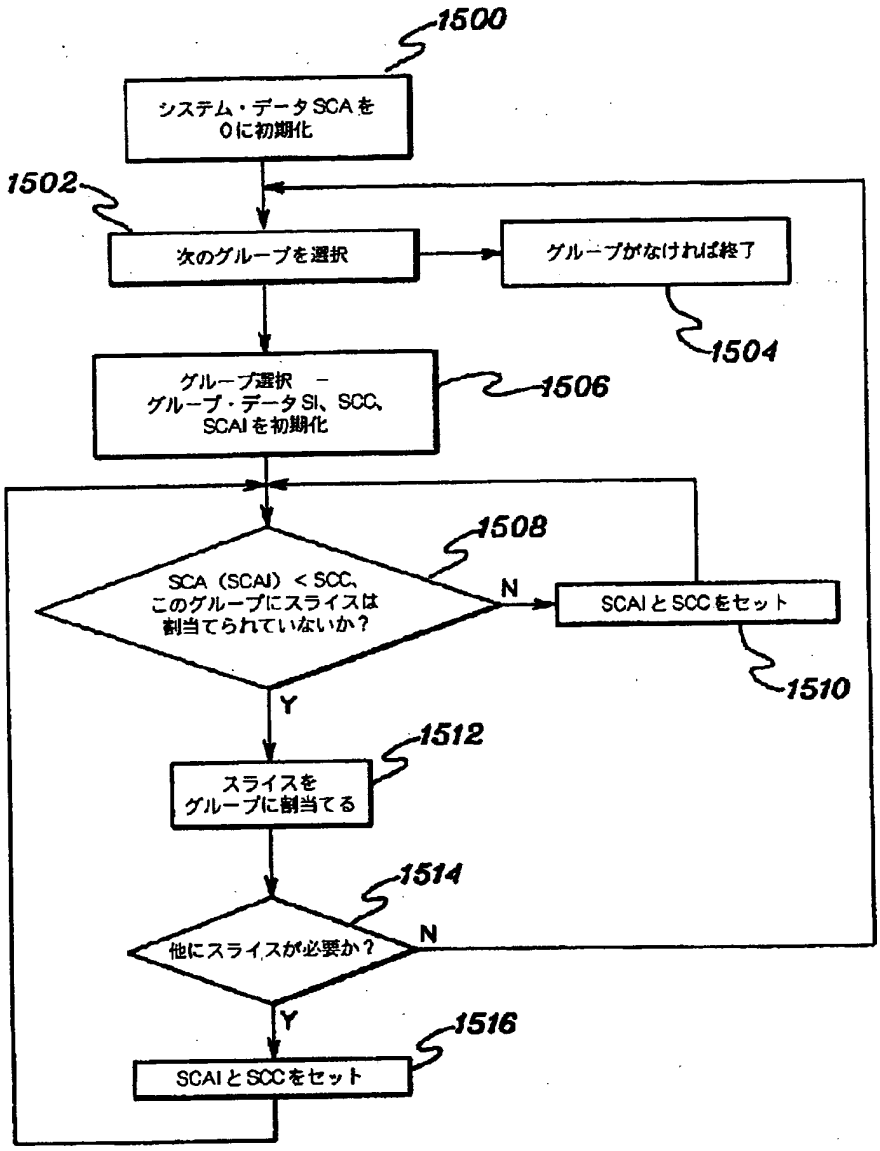
【 図１４ 】


1400
1402
他に最大値のグループありか？
N
終了
A
Y
1404
スライス＞１または サービス＞最大値か？
N
Y
1406
RSA＝最大値、LSA＝０をセット
すべてのIMPでLS（IMP）、TS（IMP）を計算
IMP＝最大値をセット
1408
B
1410
RSA ＞ 0 ?
N
Y
1412
TS（IMP） ＜ RSA ?
Y
N
1414
LSA＝LSA＋LS（IMP）をセット
1416
LSA＝LSA＋RSA＊LS（IMP）/TS（IMP）をセット
C
D

【図15】


A
B
C
D
1422
次に小さい
IMPをセット
1418
RSA = RSA – TS (IMP) をセット
Y
他にIMPがあるか？
1420
N
1424
最大値未満か？
N
Y
1426
ローカル・システムの
作業がグループの最大IMPか？
N
Y
1428
LSA = LSA + 最大値未満のサービス / 高IMP作業のあるシステム
1430
LSAをキャップ・スライスに変換

【 図１６ 】



フロントページの続き

(72)発明者　ピーター・バーガーセン・ワコム
アメリカ合衆国12590、ニューヨーク州ワッピンガーズ・フォールズ、ワイルドウッド・マナー　17ビィ